OPTIMAL UAV PATH PLANNING FOR TRACKING A MOVING GROUND VEHICLE WITH A GIMBALED CAMERA

THESIS

Riley A. Livermore, 2LT, USAF

AFIT-ENY-14-M-33

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENY-14-M-33

OPTIMAL UAV PATH PLANNING FOR TRACKING A MOVING GROUND VEHICLE WITH A GIMBALED CAMERA

THESIS

Presented to the Faculty

Department of Aeronautics and Astronautics

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Aeronautical Engineering

Riley A. Livermore

2LT, USAF

March 2014

AFIT-ENY-14-M-33

OPTIMAL UAV PATH PLANNING FOR TRACKING A MOVING GROUND VEHICLE WITH A GIMBALED CAMERA

Riley A. Livermore
2LT, USAF

Approved:

//signed//	10 Mar 2014
Richard G. Cobb, PhD (Chairman)	Date
//signed//	10 Mar 2014
David R. Jacques, PhD (Member)	Date
//signed//	10 Mar 2014
John M. Colombi, PhD (Member)	Date

Abstract

This research develops a path planning algorithm that autonomously controls a UAV to provide convoy overwatch. The scenario investigated involves the convoy team hand launching a small UAV, outfitted with a gimbaled camera, and commanding the aircraft to fly at a constant altitude and standoff distance from the convoy. The research aims to determine the optimal path to fly the UAV for convoy overwatch and demonstrate that ability onboard a real UAV through flight testing. The optimization algorithm determines the best path to fly through developing a cost function that minimizes the control effort of the UAV and the deviation from a desired slant range. The limited processing power of the autopilot prevents the implementation of this optimal controller onboard the flight test UAVs. Therefore, a heuristic-based algorithm was developed and implemented on the autopilot to approximate the optimal solution. In flight test, the UAV successfully tracked a moving ground vehicle by continually placing the UAV's loiter point directly above the ground vehicle's current location. This method was called the "follow-me" mode and provided the baseline for real-world UAV convoy overwatch. The follow-me mode resulted in a cost function value that was 113 times greater than the optimal path. Through an in-depth analysis, the heuristic-based approach reduced this ratio down to only 7.5 times greater than the optimal path. Due to the small flight test sample size, more flight tests are recommend before any general conclusions are made regarding the performance of the heuristic-based approximation. Regardless, the data collected shows tremendous promise for improving autonomous UAV performance through optimal control techniques.

Acknowledgements

First and foremost, I must thank my Lord, Jesus Christ, for blessing me with this incredible thesis research topic and the opportunity to get my master's degree.

My utmost appreciation goes out to Dr. Rich Cobb. Thank you for your patience during this whole process, your flexibility to meet with me at a moment's notice, and for putting the time and effort into creating a worthwhile product.

Thanks to my committee members, Dr. Dave Jacques and Dr. John Colombi. Both of you were instrumental in the flight test process and your knowledge and advise were critical to all of the major facets of this research. Additionally, thanks to Mr. Rick Patton and Mr. Don Smith for providing the technical assistance and flight test resources necessary for testing. You both served important roles and were pivotal to the success of the flight tests.

Thank you to Capt Charlie Neal for being a great partner in crime. Without your help and expertise, I would have never been able to successfully flight test an autonomous convoy overwatch. Thanks for all of the long hours you spent to make this research a success.

A special thanks to my sponsor AFRL/RQQA and Dr. Derek Kingston for funding my research and providing me with an interesting and relevant topic.

Thanks also to Col Angie Suplisson and Col Nate Smith for taking the time to try and get my problem to run in GPOPS. Your knowledge of optimal controls was invaluable and I deeply appreciate you investing your time to help me with my research.

Lastly, thanks to all my friends and family who have supported me throughout this process. Your investment into my life inspires me to continually pursue greatness.

Riley A. Livermore

Table of Contents

Page

List of Figures

List of Tables

List of Symbols

OPTIMAL UAV PATH PLANNING FOR TRACKING A MOVING GROUND VEHICLE WITH A GIMBALED CAMERA

I. Introduction

1.1 Motivation

Unmanned Aerial Vehicles (UAVs) constitute an ever increasing, rapidly developing tenet of the American military's air power. These remotely piloted vehicles provide many unique advantages not afforded by manned flight. The abilities to remain airborne for days and perform missions unsafe for manned aircraft are just some of the benefits of UAVs. As the technology surrounding UAVs has improved, there has been a renewed initiative to incorporate unmanned platforms into more and more mission phases. Since 2002, the number of operational UAVs has increased over 40 fold and now UAVs comprise an astounding 41% of the military's aircraft inventory [9]. Additionally, the amount of money invested into UAVs has skyrocketed from $284 million in 2000 to $3.3 billion in 2010 [9]. The value of unmanned assets is undeniable and the Department of Defense (DoD) continually seeks out ways to safely use and apply this technology to the various DoD mission needs.

The Office of the Secretary of Defense published the *Unmanned Systems Roadmap 2007-2032* [18] to highlight the military's plan for UAV development over the next twenty-five years. This document enumerates several military needs, with the intelligence, surveillance and reconnaissance (ISR) mission being foremost. The second need highlighted by the document is the target identification and designation mission. Currently, both of these tasks are possible and are being executed in a real-world, combat environment [19]. The question however remains, how can the current technology improve to increase mission effectiveness? The answer involves increasing the level of UAV autonomy. Today's limiting factor for a UAV is not the platform itself, rather

it is the level of control required by the pilot and sensor operator. Currently, most UAVs are controlled directly by an operator via a ground control station. Bolstering the level of autonomy not only reduces the crew size of these vehicles, but also strengthens the UAV's ability to deliver vital, real-time intelligence to the warfighter.

The title *Unmanned Aerial Vehicles* broadly describes all remotely piloted aircraft regardless of their size, weight, operating conditions or mission. In this research, the UAVs used are hobbyist airplanes that weigh between 5lbs and 15lbs. While the UAV is integral to unmanned operations, it is considered incomplete to neglect the ground station component of UAV operations. Therefore Small Unmanned Aerial System (SUAS) gives a more complete and thorough understanding of what is truly being tested and deployed. The subtle difference between the aircraft itself and the entire system may seem nuanced, but in reality it is an important distinction. The systems interface between the aircraft, ground control station and the user is a vital link for operations. Increasing the level of autonomy simply shifts the workload from the human component to either the aircraft autopilot or the ground control station. The goal of a "smarter" SUAS is not to completely extract humans out of the loop; instead greater autonomy allows the SUAS to execute the lower-order tasks, without human interaction, therefore maximizing the human potential in a combat environment.

1.2 Problem Statement

Currently, UAVs are being used to provide convoy overwatch on the battlefield. The benefit of using UAVs is that they provide a cost effective and capable ISR platform. The downside is that UAVs require a human operator to fly the aircraft and steer the onboard camera, instead of participating in the more crucial aspects of convoy operations. In the event that the convoy is attacked, the operator must either abandon the SUAS to help or compromise his safety by continuing to operate the aircraft. The Air Force Research Laboratory (AFRL) is interested in automating the tracking and surveillance of convoys using a SUAS platform. Unfortunately, SUASs

have a limited flight endurance because of their small size, which can be problematic for longer convoy routes. AFRL's intent is to automate the SUAS control and camera operation to supply the convoy commander with live, full motion video of the convoy, while maximizing the UAV's time aloft. For example, picture a convoy protection scenario where the SUAS is tasked with monitoring the lead vehicle. The point of interest (POI) constantly changes and is characterized by the behavior of the lead vehicle. The convoy updates the SUAS with its GPS coordinates to assist with tracking and pointing the camera. The ground vehicle is able to change its heading and vary its speed. Additionally, the SUAS is outfitted with a 360° pan and 90° tilt gimbaled camera located on the bottom of the aircraft fuselage. Using the target's position, as well as the wind speed and direction, the SUAS autonomously orients the aircraft and camera to minimize the control effort while maintaining a desired standoff distance from the POI.

The task of UAV convoy surveillance can be separated into the three functions of video surveillance, target location and path planning. Each function has a role that is unique to the tracking task and therefore irreplaceable. The level of responsibility that each function plays in the tracking task depends on the type of UAV employed. For example, a UAV with a fixed camera places more burden on the path planning function than a UAV equipped with a two-axis, gimbaled camera. Likewise, the computational burden of developing the UAV path can be simplified with a robust ground vehicle location function. Not only can the functions vary in complexity and effort, but they can also be accomplished in several different ways. Achieving these desired functions may require several hardware components which may be onboard the UAV, on the ground station or provided from another source. Regardless of how the functions are achieved, their completion is required for a UAV to track a moving ground vehicle.

The thrust of this research is developing the path planning portion of the tracking mission. At its root, the path planning function creates the path that the UAV flies in response to the location and heading of the ground vehicle (POI). The output

of the path planning function is a series of waypoints that the UAV flies. A cost function is developed based on a set of user requirements for autonomous UAV convoy overwatch. This cost function is solved using optimal control techniques to develop the desired path based on the a priori knowledge of the ground vehicle's location for all time. Optimal control is used because it represents the theoretical "best" solution and provides a point of comparison for other methods. Investigating a look ahead time solution, determining the weights for the multi-objective cost function and choosing the solver's mesh frequency are a few of the research questions that must be answered to fully develop the path planning function. This thesis research develops, implements and evaluates the optimal solutions and heuristic-based approximations generated for achieving autonomous UAV convoy overwatch.

1.3 Scope and Assumptions

This research aims to accomplish two objectives. The first is to develop an optimal flight path algorithm for any given ground path and validate it with computer simulations. More specifically, the algorithm determines the smallest control inputs required for a UAV to maintain a desired slant range while tracking a ground vehicle. The second goal is to implement the optimal path algorithm onboard the UAV to demonstrate a real-world autonomous tracking capability. Unfortunately, the computational cost of the optimization process is too high to fully implement onboard the current test UAV. This means that sub-optimal, heuristic-based control must be implemented and tested. The second goal then is to demonstrate a real-world autonomous target tracking functionality and determine how close this heuristic-based approach is to the optimal path that corresponds to the same ground vehicle path.

Simulating the UAV path planning function requires a mathematical model of the UAV. Additionally, the simulated environment must have similar features that are comparable to the real-world scenario. Assumptions are essential to create a simple, efficient mathematical model. The difficulty lies in determining what can be assumed away and what must be considered. Mathematically accounting for every specific

detail creates an unnecessarily complex system that takes an inordinate amount of time to solve. Conversely, an over simplified problem may be efficient but it produces unrealistic results. The dichotomy between accuracy and efficiency, especially in the flight test realm, is a constant struggle. It is imperative to have realistic assumptions, because the results of the computer simulations are compared to actual flight test data. The assumptions below attempt to find a common ground between these two conflicting values and give results that are simultaneously fast and accurate.

1. UAV modeled as a point mass
2. UAV flies at a constant altitude
3. UAV only makes level, coordinated turns
4. UAV always knows ground vehicle coordinates
5. Gimbaled camera always points at the ground vehicle
6. Ground vehicle travels at a constant elevation and does not exceed the UAV's groundspeed
7. Earth is assumed flat

Each assumption helps categorize the problem into a feasible solution, while maintaining the problem's integrity. These assumptions are fully explained and justified in Chapter III.

1.4 Thesis Outline

This first chapter provides an introduction to the motivation and objectives of the research. Chapter II lays the foundation with a comprehensive review on previous research topics surrounding UAV path planning and tracking. Additionally, Chapter II briefly overviews optimal control problems and their solution techniques, as well as flight testing specifics. Chapter III provides the methodology of the thesis research. This chapter presents a step-by-step process of transforming between multiple coordinate frames, developing the optimal control problem and explaining

the flight tests used to develop the heuristic-based approximation. Chapter IV introduces the optimal solution by discussing the development of the "full path" and "look ahead" algorithms. Chapter V uses the optimal paths to help develop the heuristic-based approach. The flight test results and final comparison of the optimal path and heuristic-based approximation are shown and analyzed. Finally, Chapter VI summarizes the research, makes conclusions and provides insight for future improvements.

II. Background

This chapter outlines the background information required to perform the thesis research and can be divided into two sections: 1) literary review and 2) flight test. The literary review provides a comprehensive look at previous UAV research regarding stationary and moving vehicle tracking, optimal path generation and wind effect modeling. The flight test section introduces the aircraft, autopilot, ground station components and other flight test related items.

2.1 Literary Review

A vast set of unique challenges come with following a moving ground vehicle with a gimbaled camera, while in the presence of wind. The literature review focuses on UAV related research that pertains to the autonomous UAV overwatch scenario. The literature review focuses primarily on developing the path planning function, but also includes different approaches to the video surveillance and target location functions. The various journals and theses selected delve into several topics that deepen the understanding of autonomous UAV tracking. These topics can be split into the categories of stationary target observation, mobile target tracking, optimal path generation and heuristic-based approximations and wind effect modeling.

2.1.1 Stationary Target Observation. A significant body of work exists for using UAVs to survey stationary targets. Understanding the dynamics required to accomplish this are fundamental for the ground vehicle tracking problem. Stolle and Rysdyk [22] investigated the generation of desired paths for observing a stationary target. The pinnacle development of their research was a control law that continually sought to maneuver the UAV from its current path to a desired path. They named their solution to this process the "Good Helmsman". Relative course heading and cross track error were the two variables the Good Helmsman used to determine how the UAV maneuvers to its desired path. Heuristic-based approaches were used to "fine-tune" the weightings of these parameters to ensure desired path convergence

within the UAV's physical constraints. The control law ran onboard the autopilot and gave real-time commands to maintain the desired path.

Rysdyk [21] further refined the Good Helmsman concept by using differential geometry to model path observance. The added depth provided in this research is crucial for observing a stationary target; however, this process breaks down when the target is allowed to move. The foundation of Rysdyk's research is valuable because it deals with deviation from the desired course heading and reacquiring the desired path. This concept could translate to a moving target because the desired course to fly is based on the ground vehicle's current and future locations. In both papers, Rysdyk did not consider any optimal approaches in his solution methodology. The goal of Rysdyk's research was to develop a real-world, implementable algorithm that observed a stationary target with a limited range, gimbaled camera. Understanding these path planning algorithms are integral to the thesis research herein and give insight for developing both the optimal and heuristic-based approaches.

Farrell [7] also developed a path planning function by generating waypoints to survey a static point of interest. He created a real-world, software module that interfaced with the UAV's control system to allow the user to interject unplanned commands that altered the UAV from its pre-programmed route. These UAVs had fixed orientation, forward and side mounted cameras and were capable of being hand launched in remote operating environments. Farrell investigated three different potential scenarios where the command override might be applied. The scenarios considered an operator who desired to gain further intelligence on a POI and instructed the UAV to orbit a given point. The UAV orbited in a manner that kept the POI in the field of view (FOV) of the side mounted camera. The second scenario featured the operator instructing the UAV to make a single pass over the target so that it was captured in the forward camera's FOV. Lastly, the operator had the UAV perform a single pass by the target at a specified offset for POI observance using the side camera. The path planning function of Farrell's waypoint generation routine considered all three of these scenarios.

While the application of Farrell's work may not seem helpful to the tracking mission, the applicability to this research effort comes from how he executes the mission. Farrell's waypoint generation concept is useful for this thesis. He created a series of waypoints for the UAV to fly that maximized time on target, in a constant wind environment. Although the intent and constraints are different in this thesis, the real-world solution must feature some sort of waypoint generation algorithm.

2.1.2 Mobile Target Tracking. Transitioning from static target observation to following a moving vehicle introduces numerous challenges to the path planning process. As the target moves, the UAV must dynamically account not only for its behavior but the ground vehicle's as well. The driving goal for the majority of the dynamic models is to maintain a set standoff distance from the moving vehicle.

Lee et al. [16] investigated a strategy for maintaining a given standoff distance from a moving ground vehicle. In their problem formulation, the UAV had a fixed airspeed and the algorithm accounted for a variable ground vehicle velocity, as well as disturbances from the wind. Depending on the speed of the ground vehicle, the UAV either flew in a sinusoidal pattern or loitered. The ratio between the UAV's ground speed and the vehicle's speed distinguished between which mode was used. Lee et al. defined the threshold, between these two modes, as a 3:1 ratio of UAV to ground vehicle speed. At faster ground vehicle speeds, the UAV varied the amplitude of its sinusoidal path to maintain the same horizontal displacement from the ground vehicle. When the velocity ratio was greater than 3:1, the ground vehicle's speed was sufficiently slow to justify loitering to maintain a standoff distance.

The authors do not use optimal control methods in the paper and only focus on maintaining a prescribed standoff distance. Regardless, the loiter and sinusoidal modes provide insight into possible behaviors that the optimal solution might share. The optimal solution should feature times where the UAV loiters or flies sinusoidally, depending on the speed of the ground vehicle. The algorithm presented by Lee et al.

is valuable because it provides a simplistic method to minimize the UAV's deviation from the desired standoff distance.

In a collaborative effort, a group from MIT investigated a framework for accomplishing a cooperative search, acquisition and track (CSAT) task [10]. Their goal was to create a decentralized network that empowered each individual UAV to make decisions and perform the required task. MIT's research was built upon a number of advances in UAV development and smartly used several UAVs to find and track multiple ground vehicles in a given space. In their problem formulation, the position and number of ground targets was initially unknown. The UAVs were given a specified search space with probabilistic locations of targets and were assigned to both search the space, as well as track any targets found. The tracking of all known targets was balanced against searching the area for possible new targets. The UAVs used Kalman filters to predict where the known targets would be in the future. Based on the confidence of their ground vehicle path estimates, the UAVs were able to temporarily abandon their tracking task and search for new targets. This whole process was achieved independently, by each UAV, through a variety of hardware devices located onboard.

The work done by MIT provides a great example of how the video surveillance, target location and path planning functions are tightly interwoven for UAV tracking. In fact, the hardware that performed the target location also drove the video surveillance. These two functions worked in tandem to provide the necessary information for the planning module to decide the flight path. As the UAVs searched the space, they were able to use the onboard gimbaled camera to locate the targets. Once the targets were located, the UAVs followed them using optical tracking to keep the target in the FOV. Optical tracking effectively combines the functions of target location and video surveillance by using the images from the video to locate the coordinates of the ground vehicle relative to the UAV. This complex process has great promise for future applications and provides insight for this thesis research.

Yoon et al. [24] uniquely accomplished the tracking problem with a pair of autonomous UAVs by using a different coordinate system. They represented the motion of each UAV using the concept of a spherical pendulum. The UAV's equations of motion were translated from Cartesian to spherical coordinates. The ground vehicle was placed at the center of the spherical pendulum, which oriented the UAV's motion relative to the target. Their main goal was for at least one UAV to maintain a desired standoff distance from the moving ground vehicle. A Lyapunov candidate function was used to determine the stability of the UAV's path planning function. Before the coordinated UAV tracking functionality was demonstrated, the viability of the spherical pendulum model was examined in a single UAV tracking scenario.

For the single UAV tracking mission, the swing angle (ϕ) was fixed and the desired circling rate ($\dot{\theta}_d$) was determined by the standoff distance and circling velocity. In the single UAV case, the aircraft successfully followed the moving target, although it did not tightly maintain a constant standoff distance. The large standoff deviations were a by-product of using the method described in the research. The method used for the single UAV target tracking was adapted to include two UAVs. For multiple aircraft, the phase angle separation between the two UAVs was also considered. The acceleration commanded had to comply with the intended standoff distance, while simultaneously considering the location of the other UAV. The two UAVs were able to maintain a phase angle separation of 90° while both following the target. Even with two UAVs, neither aircraft was able to consistently maintain the desired standoff distance. These deviations were consistent with the single UAV scenario because the same tracking algorithm was applied in both the single and paired UAV tracking cases. The researchers did not mind the standoff distance error in their research and were more concerned with the utility of the model for a simple autonomous overwatch scenario.

Using a spherical pendulum to evaluate motion of the UAV, relative to a moving ground vehicle, allowed the authors to incorporate complex equations of motion and control schemes into a simple set of equations. This generation of equations dif-

fers from the method used in this thesis formulation. However, understanding their formulation provides another unique way of interpreting the tracking problem. Unfortunately, most of the work done by Yoon et al. fails to evaluate the utility of using a camera for tracking or compensate for the additional challenges involved with video surveillance. Additionally, their path planning function does not incorporate any notion of optimality. The utility of their method arises in its possible applicability to a heuristic-based approach. The value of their approach is its simplicity, speed and efficiency; making it a viable possibility for the heuristic-based approximation model, which will be investigated herein.

The work done by Quigley et al. [20] provides a comprehensive analysis of the three functions required to perform UAV overwatch. They discuss how the video surveillance, path planning and target location functions are executed in a Mini-UAV platform.

The first portion of their research focused on executing the video surveillance mission. For any UAV tracking platform there must be a trade analysis done with both fixed and gimbaled cameras. They concluded that a successful tracking platform must incorporate a gimbaled camera. Using a full rotation, 360° pan camera on their UAV platform provided the maximum flexibility for video surveillance. The gimbal setup recommended by Quigley et al. shares numerous similarities with the one used in this thesis work. Additionally, the azimuth and elevation angle equations and the lessons learned from their research provides a great starting point for this thesis research. Using a full rotation, gimbaled camera eases the burden of the path planning function by increasing the number of path possibilities, while constantly maintaining the target in the FOV.

The path planning algorithm was derived from the mathematical function known as a Hopf bifurcation. The Hopf bifurcation solution produced a spiral trajectory that converged to a stable limit cycle, which in this case was a circle. When inside the limit cycle, the trajectories spiraled outward towards the circle and points outside the

limit cycle would spiral inwards to the circle. In their path planning algorithm, the center point was collocated with the location of the ground vehicle. To enforce a given standoff distance, the limit cycle radius was set at the preferred standoff distance for the UAV. As the ground vehicle moved, the center point and the limit cycle moved accordingly. The UAV constantly flew towards the desired standoff from the ground vehicle and effectively created a simple, implementable path. This concept does not promote the notion of optimality; however, it provides a simplistic, accurate method for a heuristic-based solution.

The target location function featured a number of different methods. For the moving targets, the location of the ground vehicle was sent to the UAV at a frequency of 2 Hz. For stationary targets, Quigley et al. developed an interface that allowed the user to triangulate the target's position through a series of target lines. As the UAV flew around, the onboard camera continuously pointed at the target. The user then created a series of target sight lines at various points in the path that intersected at the location of the target. Using this method, the UAV was able to autonomously begin to loitering around the target at the desired standoff. Quigley et al. acknowledged the importance of target location, although they did not develop robust methods of determining the current ground vehicle position.

Jones [12] successfully demonstrated autonomous vehicle tracking with a UAV called the MLB Bat. This UAV had a wingspan of 6ft, weight of 15lbs and was outfitted with a two-axis gimbaled camera. During the flight test, the UAV was bungee launched from the convoy vehicle and commanded to loiter above the vehicle at a certain altitude and standoff distance. Throughout the entire flight, the UAV was constantly updated with the convoy's GPS coordinates. The operator had a choice between commanding the UAV to perform advance route surveillance or follow the convoy itself. In either case, the UAV pointed the gimbaled camera at the intended GPS location. Additionally, Jones incorporated a variable throttle functionality, which differs from the fixed throttle assumption made in this thesis. During his flight tests, the UAV could change its heading and velocity to keep it in the ap-

propriate position relative to the convoy. Varying the throttle simplifies the path planning process, but it comes at the cost of endurance. Jones successfully flight tested the MLB Bat by demonstrating what he called the "zero operator" mode for convoy protection.

The three functions for convoy operations are not explicitly enumerated by Jones, although he specifically addresses them in the document. His approach to video surveillance, target location and path planning are all nearly identical to this thesis efforts. The optimal control emphasis of this thesis marks the notable deviation from Jones' research. The necessity of a heuristic-based approach is not to simply achieve autonomous convoy overwatch; rather, it is to best approximate the optimal solution using sub-optimal methods. This work done by Jones provides a great foundation to build a heuristic-based approach for the convoy overwatch scenario.

2.1.3 Optimal Path Generation and Heuristic-Based Approximations. Describing a path as optimal infers that the generated trajectory represents the best possible path, according to a pre-defined cost function. This process is oftentimes computationally intensive, and therefore appears, in the literature, more in pre-processing or post-processing than in real-time applications. The optimal path planning approach defines the work done by Zollars [25], who found the minimum time to intercept a ground vehicle in a constant wind environment. Zollars used a SUAS with fixed, side and forward looking cameras for his research. He evaluated the utility of optimal control for three different scenarios.

First, Zollars looked at the minimum time to intercept the moving vehicle in an open environment. The cost function minimized the time required for the UAV to capture the vehicle in the forward looking camera's FOV. In this first scenario, Zollar's used a Dubin's path approximation as the initial guess for the optimizer. Dubin's path is a construct that claims the shortest distance between an initial and terminal position and heading consists of two maximum turns and a straight line [25]. The presence of wind precluded the Dubin's path from being the optimal solution, but

the Dubin's path still provided the optimizer with a good initial guess nonetheless. His dynamic optimization problem solved a free final time, fixed final state optimization problem for this first scenario.

The second scenario determined the optimal path to intercept a moving target in an "urban canyon" environment. Envision a city metropolis with numerous buildings that could possibly interfere with the free space solutions developed in the first scenario. For this environment, the Dubin's path generated in the first model did not apply to the urban canyon environment. Instead, a new cost function was created and a modified Dubin's path was used to force the UAV to converge to a straight line solution. The UAV followed a series of straight line solutions until it reached a certain proximity of the target. This solution forced the UAV to a path that allowed it to intercept the target quickly, while avoiding obstacles. Similar to the first scenario, this problem also featured a free final time, fixed final state optimization problem. The main difference between the two solutions was the inclusion of the path constraints caused by the urban canyon.

The last scenario encompassed target acquisition and continual tracking. This objective was broken into three phases: target acquisition with a forward sensor, transition to orbit and surveillance with a side mounted camera and finally maintaining the ground vehicle in the side camera in the presence of wind. The final portion of Zollar's research most closely resembled the convoy overwatch effort of this thesis. The path planning piece was Zollar's main focus and its emphasis was paramount due to the UAV's fixed cameras.

Incorporating Zollar's algorithm in a real-world environment becomes problematic due to the computational cost of running the optimizer, either onboard or remotely on a ground station. In some instances, it took upwards of 15 seconds to create the optimal path. In a real-world scenario, the UAV must continue to fly as it computes the path. This means that once the solution is generated, it is already outdated because the UAV and ground vehicle are in completely different locations.

Terning [23] sought to determine a faster, sub-optimal method for accomplishing the same goals of Zollars. Terning acknowledged that the optimal solution was not feasible in a real-world environment due to the computational cost it took to develop the optimal path. His solution incorporated an iterative process that continually updated the UAV's current waypoint. At each iteration, Terning determined the intersection point between the UAV and the moving ground vehicle and adjusted the UAV's heading accordingly. He assumed that the ground vehicle's speed and the UAV's airspeed were constant throughout the test. This allowed the algorithm, which he nicknamed "Pathmaker", to iteratively converge on the path for the UAV to intercept the ground vehicle.

Terning did not consider the UAV's actions once it acquired the ground vehicle within its field of view. He was merely concerned with intercepting the vehicle's path as quickly as possible. Although his mission objective differs from the convoy overwatch goal for this thesis, the necessary functions to accomplish the respective tasks are the same. Terning had to consider the video surveillance, target location and path planning functions in his Pathmaker algorithm. Using fixed camera's onboard the UAV decreased the burden of the video surveillance function, but increased the workload of the path planning algorithm. Additionally, the success of the Pathmaker algorithm hinged on receiving timely and accurate ground vehicle locations. Terning's framework for creating a heuristic-based approach established a good model for incorporating similar techniques for the convoy overwatch scenario.

Recent research by Kim et al. [14] used optimal control to track a moving ground vehicle with a pair of UAVs. The UAV was treated as a point mass and only operated in the horizontal plane. The UAV had two controls, heading and throttle, while the ground vehicle could also change its speed and direction. A pair of UAVs independently tracked the moving ground vehicle using a specialized onboard radar. To maximize their accuracy, the UAVs had antipodal orbits, which meant they maintained 180° of phase separation within the orbit. Next, a Kalman filter estimated the future path of the ground vehicle based on its current location and previous

behavior. The UAVs used state vector fusion to cross-reference their location and path estimates of the ground vehicle. The results from the state vector fusion defined the initial conditions that were fed into the optimizer to find the optimal path. The optimal path formulation was continually updated as the UAVs and ground vehicle marched forward in time.

The work done by Kim et al. stimulates numerous ideas for developing the path planning function. The first, and most closely applicable to this thesis, is that Kim et al. used optimal control to determine the UAV's flight path. They developed a governing set of equations of motion, identified control variables and established a cost function. The optimal path algorithm had to consider all of these factors and converge on the controls that resulted in the optimal solution. While Kim et al.'s problem formulation and cost function may differ from this thesis, their process for determining the optimal solution is equally valuable. Secondly, the optimal path planning relied on representative ground vehicle profiles. Kim et al. used a robust model to simulate the vehicle's ground path. In a real-world scenario, receiving timely and accurate coordinates of the ground vehicle's location is essential for calculating the optimal path. Choosing a representative ground path model helps make the simulations realistic and applicable to a real-world situation. Lastly, Kim et al. compared their optimally generated path with the Lyapunov vector field solution flight tested by Frew et al. [8]. In their paper they show how the optimal solution decreased the deviation error from the desired vehicle standoff and enforced the antipodal separation of the two UAVs. The process of comparing two different solutions to the same ground path is one of the main objectives of this thesis.

The focus on the video surveillance and target location functions define the main differences between this thesis research and the work done by Kim et al. Kim et al. did not address video surveillance nor did they account for the use of a video camera in their cost function. The lack of emphasis on video surveillance was overcome by the additional focus on target location. The robust target tracking algorithm featured an onboard radar to locate the ground vehicle's location and an Extended Kalman

Filter (EKF) to predict its future location. This information was then fed into the optimizer to help determine the corresponding optimal UAV flight path. This process will not be performed in the thesis work, but provides a necessary step to further increase the robustness of the algorithm.

2.1.4 Wind Effect Modeling. Wind can dramatically impact a UAVs flight path. Due to their low operating speeds, UAVs can oftentimes experience wind speeds close to their airspeed. For this reason, a path planning model must incorporate some dynamic measures to account for wind speed and direction.

Stolle and Rysdyk [22] investigated how wind affected the UAV's performance as it maintained a standoff from a stationary target. UAVs had to "crab" into the wind to maintain the desired ground path. This maneuver put sideslip on the aircraft, which reduced its overall endurance. The alternative was to allow the plane to turn into the wind, causing the UAV to depart from its desired path. This behavior was most noticeable in a direct crosswind and the resulting orientation of the plane potentially left the target outside the bounds of the gimbal limits. To further complicate the problem, the authors [22] also considered the location of the sun relative to the UAV and target. They specifically planned the UAV paths to avoid orientations that led to image degradation caused by poor sun orientation. The worst orientation occurred when the UAV was directly between the target and the sun because the image was overexposed by the direct sunlight. Therefore, the goal of the authors was to find a feasible path to fly that limited the effects of the wind and the sun, while maximizing time on target. Stolle and Rysdyk developed three feasible solutions for flight paths that avoided detrimental wind and sun regions. Each path used either a semi-circular path, elliptical path or combination of the two for creating the desired path. The real value of this research is that it incorporates realistic strategies for overcoming wind in a dynamic environment. The effects of wind are often understated, but can greatly impact the success of the algorithm, especially during a flight test. Considering only

a stationary target greatly simplifies the problem for overcoming adverse wind and environment effects.

Farrell [7] investigated maintaining a sensor aim point on a fixed target in the presence of wind. Even though the target was stationary, the presence of wind introduced a level of complexity in maintaining the target in the FOV. Farrell successfully created an algorithm that generated waypoints to keep the UAV's fixed camera pointed at the target. Most importantly, he incorporated a robust wind correction algorithm to increase the accuracy of his waypoint generation algorithm. His flight test results showed that his algorithm kept the target in the FOV at least 66% of the time as long as the wind speed was less than 25% of the UAV's cruise speed. The research effort by Farrell is useful and has many similarities to Stolle [22] and Rysdyk [21]. However, Farrell's research is not applicable once the target is allowed to move. Since his cameras were fixed onboard the UAV, it was impossible to consistently maintain a moving vehicle within the camera's FOV. For this reason, the thesis incorporates a gimbaled camera, which excludes time on target from interfering with endurance and standoff distance.

Quigley et al. [20] investigated wind effects on a UAV's tracking performance. They noticed that the UAV was capable of orbiting a moving point in winds that were 65% of the UAV's airspeed. Any winds exceeding 75% of the airspeed created situations where the UAV's heading estimation errors caused large deviations from the desired course heading. This information is important because it represents how SUASs behave in wind environments and directly applies to the flight test potion of this thesis.

2.2 Flight Test

This section introduces the specific elements required to flight test the autonomous UAV convoy overwatch. The airframe, autopilot, camera setup, ground control station, test range and ground vehicle are all introduced and their importance explained. Each one of these categories plays an integral role in the test operations.

2.2.1 Airframe. The Sig Rascal 110 (aka Rascal) (Figure 2.1) is the airframe used for the flight tests. This remote control (RC) hobbyist aircraft is a proven testbed that has been used for numerous AFIT flight tests dating back to 2006. The Sig Rascal has a high mounted wing with a tail-dragger landing gear. The aircraft has a 110" wingspan, is 75.75" long, weighs 11 pounds empty and is propelled by an electric motor [11]. The motor is powered by three, 4 cell batteries which allows for about 40 minutes of flight time. An electrically powered aircraft is advantageous for a number of reasons. The primary reason is the electric motor creates significantly lower vibration levels than a gas motor, which improves the image quality of the video surveillance. Additionally, batteries allow the UAV to operate in a variety of conditions and make the plane easier to maintain in the field. The tail-dragger setup allows for the video surveillance payload to be mounted close to the main landing gear near the Rascal's center of gravity. Locating the camera underneath the Rascal's fuselage allows for the gimbal to have an unimpeded 360° of pan and 90° of tilt. This enables the camera to always point at the target, giving the UAV increased flexibility for following the ground vehicle.

Figure 2.1: Sig Rascal 110

The primary purpose of flight test is to demonstrate a proof of concept for UAV tracking. While the Sig Rascal is not an operationally viable UAV that would be used in a combat environment, it is similar in size, weight and performance to the UAVs that would be used for convoy tracking. The Sig Rascal operates at around 15

m/s (33.5 mph) with the ability to fly up to 25 m/s (56 mph), which matches well with ground speeds that the test vehicle drives at the range. The Rascal is a capable, proven platform that provides the representative data required to prove the concept of autonomous convoy overwatch.

2.2.2 Autopilot. The ArduPilot Mega (APM) is the autopilot used for the flight test portion of the research. APM is a full-featured autopilot that incorporates all of the necessary functions for a variety of flight modes. The autopilot includes a 3-axis accelerometer and rate gyros that measure the orientation changes of the aircraft. The APM 2.5 (Figure 2.2), the specific model used for flight testing, contains a 16 MHz Atmega328 processor onboard and can control 8 different channels [1]. The APM interfaces with a RC controller, which is necessary for safety pilot oversight. The autopilot allows the gain structure to be tuned for both the inner stability loops, as well as the outer navigation loops. Tuning these gains enables the SUAS to have appropriate handling qualities while performing autonomous missions. APM also has a direct feed for GPS which allows it to fly point to point navigation. This functionality provides the necessary baseline for flight testing autonomous UAV convoy tracking.

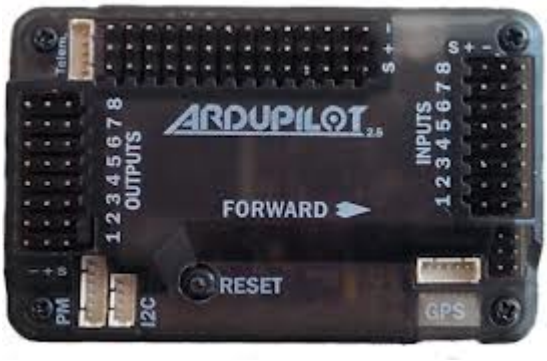


Figure 2.2: Arduino Autopilot - APM 2.5 [1]

The APM is an open source autopilot that provides the user the ability to completely adapt and customize the autopilot code to meet the specific mission needs. The autopilot code uses a C-based programming language and comes with a cor-

responding programming environment called Arduino IDE [1]. Within the Arduino IDE, all of the autopilot code is accessible to be read, modified and compiled so that any changes can be flashed to the APM board and flight tested. An open source autopilot is advantageous when it comes to incorporating complex missions, like camera directed UAV tracking. The makers of the APM have done a remarkably good job with configuration control. They have a select group of individuals who regulate what updates are released, which correlates to reliable and stable firmware.

The APM can determine the wind speed and direction using the onboard GPS. The autopilot cross references the airspeed and heading with the GPS ground track and backs out the corresponding wind conditions. While this functionality is not perfect, it provides real-time wind estimates with sufficient accuracy. The wind data is provided in the telemetry stream and is necessary for calculating appropriate optimal paths.

The ability to perform hardware in the loop (HIL) simulations is another crucial component of the APM. HIL allows the actual autopilot, in conjunction with a flight simulator, to be flown and tested on the computer. This functionality provides a safe, inexpensive environment for practicing flight operations and testing modified code. In this thesis research the autopilot code is modified to allow the UAV to autonomously track a moving ground vehicle. HIL simulations enable this updated code to be tested before it is ever flown on a real aircraft. Using HIL simulations allows the team to analyze the tracking performance of the UAV and prevents unnecessary mishaps from occurring due to coding errors. The HIL environment is also used to fly all of the test points prior to going to the field. This prepares the test team for the real flight test and gives them a better idea of what to expect when flying the actual aircraft.

AFIT has only recently started using APM autopilots for their flight tests. Previously, other teams used Procerus Technology's Kestrel [5] and Cloud Cap Technology's Piccolo [3] autopilots for their flight tests. These autopilots functioned fine within the confines of those test flights, although they are significantly more expensive

and the underlying autopilot code is not user accessible. For this reason, the APM is the clear choice to fly this tracking mission.

2.2.3 Camera Setup. The AFIT team chose the HackHD [2] camera for gathering the video data during flight test. This camera, shown in Figure 2.3, records video in 1080P high definition and is outfitted with the stock 160° wide angle lens. The lens is interchangeable, which allows for different lenses depending on the resolution and pixel density requirements. Additionally, the camera has an onboard micro SD card slot to record all video taken at 1080P resolution. This feature allows the team to record the high definition video taken by the camera and not rely on the lower quality video transmitted down to the ground station.

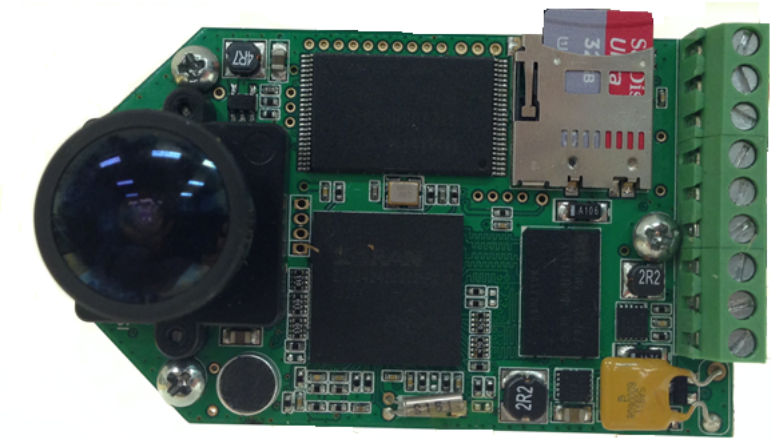


Figure 2.3: HackHD Camera

The HackHD is chosen because it has a high quality picture at a cost effective price. It is important to note that the camera is a nice feature to have, but is not vital to the test operation. The research only evaluates the optimality of the path flown by the UAV and does not have any camera related metrics. The addition of the camera allows for an in-flight sanity check that the UAV is tracking the ground vehicle.

The HackHD is attached to a pan-tilt gimbal, powered by two Hitec servos [4]. The configuration of the servos and HackHD camera is shown in Figure 2.4. The pan

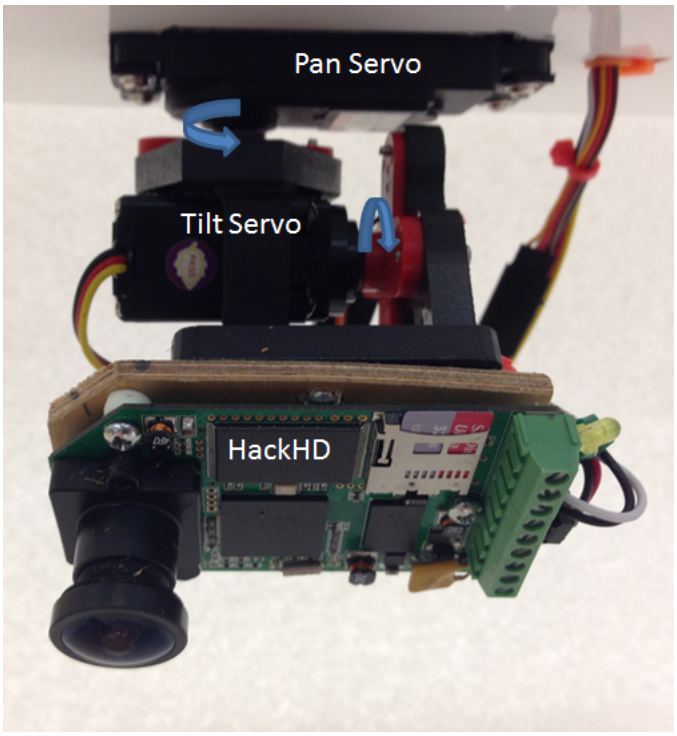


Figure 2.4: Camera Apparatus: Pan-Tilt Servos and HackHD

servo, which varies the azimuth angle, is an analog servo that can rotate 1260°. Due to wire length limitations, the pan servo is set to range from −180° to 180°. The full 360° azimuth range is still covered, although it is not continuous throughout that spectrum. Upon reaching one of the pan servo limits, the camera must rotate 360° in the opposite direction to continue the path. This gimbal specific issue does not affect the UAV's flight path and is only an inconvenience in the rare case where the azimuth angle equals 180°. The tilt servo, responsible for changing the elevation angle, is a digital servo that can rotate 180°. The gimbal limits in the autopilot are set from 0° to 90° which keeps the elevation angle within the desired servo range.

2.2.4 Ground Control Station. The ground control station (GCS) allows the UAV to operate in an autonomous manner. The GCS runs on a laptop and uses the Arduino customized operating software MissionPlanner (shown in Figure 2.5). This robust interface allows the UAV to complete nominal tasks, such as loitering

and waypoint navigation, but also allows for customized code to be implemented and run real-time.

There are three main frequencies used during flight testing. The 900 MHz band is the data band that the GCS sends and receives commands to and from the autopilot. The MissionPlanner interface relays the real-time UAV actions down to the laptop so that the grounds station operator (GSO) has constant awareness of the UAV's actions. The GCS collects and stores the telemetry from this frequency allowing the team to later access it for post processing and evaluation. The 2.4 GHz band is reserved for the safety pilot's channel. The GCS does not directly control this link but is cognizant when the safety pilot has command of the UAV. This link is important because it allows for manual override in the event of an autopilot malfunction. Having a designated safety pilot frequency allows for the team to safely test the UAV's ability to perform autonomous target tracking. The 5.8 GHz frequency is the last link used by the GCS. The video transmitter broadcasts over this frequency and the GCS can stream the live video from the HackHD to the MissionPlanner software. All three of these frequencies are used in flight test. Having distinct frequencies for the video surveillance, path planning and safety pilot control allows for multiple commands to be sent to the UAV simultaneously without interfering with each other. Additionally, MissionPlanner has the capability to perform HIL simulations. This is important because it stimulates understanding and mission planning in a controlled environment so that the team is better prepared for flight test.

Aircraft flight characterization and ground vehicle tracking are the two main categories that are flight tested. While characterizing the aircraft's flight performance, the GCS remains stationary and the team executes the appropriate test procedures. Once the ground vehicle tracking portion is ready for testing, the GCS is placed inside the ground vehicle. The UAV is not tracking the vehicle itself, rather it is trying to maintain a standoff distance from the moving ground station. Having the ground station in the vehicle collocates the GSO with the driver of the ground vehicle. This is beneficial for testing, as well as ensuring the safety and interoperability of the UAV

Figure 2.5: MissionPlanner GCS software

for the given tracking mission. The test setup is intended to mimic the real-world autonomous UAV convoy overwatch scenario. In the operational implementation, the GCS is in the convoy vehicle and the convoy is transmitting its location to the SUAS. Therefore, placing the GCS in the ground vehicle makes sense from both the test and real-world perspectives.

2.2.5 Test Range. All flight tests are conducted at Camp Atterbury Joint Maneuver Training Center, IN. This Army installation has restricted airspace which allows AFIT to test there without having an FAA issued certificate of authorization. Camp Atterbury has extensive ranges and multiple launch sites for UAVs, which make the location prime for flight test. AFIT has access to Camp Atterbury's main runway, Himsel airfield (11II), as well as the SUAS airstrip. Both facilities have their respective advantages and are used during flight test. Himsel airfield has a long, paved runway that is easy for takeoff and landing and provides an ideal location for gain tuning and introductory test items. The SUAS airstrip does not have a paved surface, which increases the takeoff and landing difficulty. However, the proximity of

the SUAS airstrip to the test range makes this a favorable location for flight testing the UAV convoy overwatch.

The range has numerous roads that allow for simulating real-world convoy operations. Figure 2.6 shows a view of both the Himsel runway and the SUAS airstrip launch points, the transit paths to the range and the ground path driven to simulate a convoy path. The path has a "figure 8" configuration that features both left and right turns, as well as some longer straightaways. The variation in the path helps stress the path planning function to better compare the optimal solution with the heuristic-based approximation. Ideally, creating a path planning function for this path should generally apply to less complex ground paths.

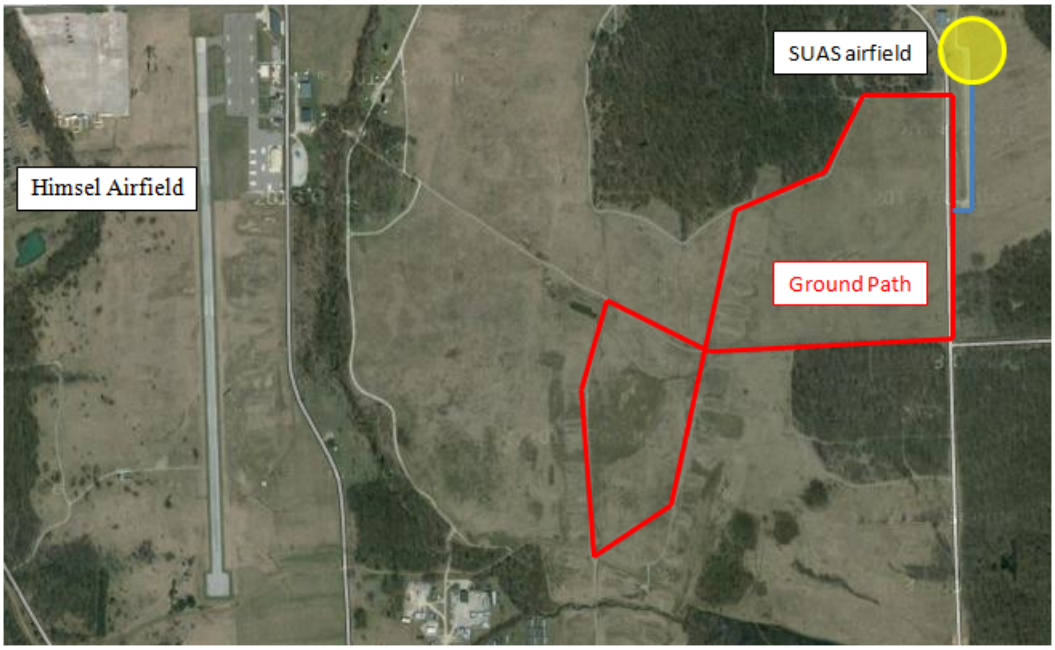


Figure 2.6: Camp Atterbury Test Range

The test team conducted three separate flight tests at Camp Atterbury. The dates for the tests were 19-21 August 2013, 21-23 October 2013 and 5 December 2013. The test dates were strategically planned to allow the team to exercise an iterative approach to the heuristic-based approximation of the optimal path. Chapters III and IV discuss the test setup, execution and the results of each individual flight test event.

2.2.6 Ground Vehicle. The ground vehicle paths are generated through driving a series of road networks at Camp Atterbury and recording the GPS coordinates. The $3-411^{th}$ company at Camp Atterbury graciously allowed the test team to use a High Mobility Multipurpose Wheeled Vehicle (HMMWV), or a Humvee as it's more commonly known, as the primary ground vehicle. Using a Humvee, shown with the test team in Figure 2.7, satisfies multiple testing requirements. The first requirement is ensuring the safety of the test team and the aircraft. All of AFIT's flight tests require a safety pilot for observance and aircraft recovery. For the tracking test, placing the safety pilot in the ground vehicle is the ideal location because it gives the safety pilot the best view of the UAV at all times. To ensure the safety pilot's safety, the ground vehicle must have a secure place to sit, while also having an open top for maximum visibility of the UAV. The Humvee presents an appropriate solution that meets both criteria. Additionally, the Humvee is perfect for the convoy overwatch because it is the exact type of vehicle that is driven in convoys.

This chapter presented a literary review of prior UAV research and covered the different components specific to the flight test. The literary review focused primarily on developing the path planning function and included a look at stationary and moving vehicle tracking, optimal path generation and wind effect modeling. Lastly, the UAV, autopilot, camera apparatus, ground vehicle and test range are all discussed. All of these items each contribute in a unique way towards the accomplishing the overarching goal of autonomous UAV convoy overwatch. The background information presented in this chapter lays the foundation for developing the specific problem of this thesis research. Chapter III takes this background information as well as the assumptions and goals from Chapter I to create the specific methodology for solving the UAV autonomous tracking problem.

Figure 2.7: Test Team with the Humvee

III. Methodology

This chapter details the process used to develop the optimal solution, as well as the methodology for testing the sub-optimal, heuristic-based solutions. An understanding of coordinate frame rotations is required to derive the necessary equations of motion. These equations are used to determine the states and controls used in the dynamic optimization solution. The cost function is created based on user requirements and impacts the characteristics of the optimal path. The output of the cost function is a scalar value that, when minimized, yields the optimal path. Lastly, the flight test section discusses how the three autopilot parameters of loiter radius, loiter range and lead time are used to approximate the optimal path.

3.1 Coordinate Frames

The three coordinate frames used are the inertial, body and gimbal frames. To successfully track a moving ground vehicle in the inertial frame, information from the body and gimbal frames must be translated to the inertial frame. A common reference frame is required to compare the UAV, camera and ground vehicle actions.

3.1.1 Inertial Frame. The inertial frame defines the unmoving reference frame that is fixed to the Earth. Due to the small distances traveled by the UAV, relative to the Earth's radius, the surface of the Earth can be assumed to be flat (Assumption 7). By assuming a flat Earth, latitude, longitude and altitude accurately describe the axes in this reference frame. This greatly simplifies the math and allows the inertial reference frame to have intuitive and easy to understand coordinate system. Typically, the inertial frame uses a North East Down (NED) coordinate system. This coordinate system follows the right hand rule where the positive x direction represents North, the positive y direction represents East and the positive z direction is down towards the center of the Earth. The inertial frame is chosen to be the common frame that the UAV body and camera gimbal are translated into for comparison. To accomplish these transformations, the body and gimbal frames must be defined with respect to the inertial frame.

3.1.2 Body Frame. The body frame is fixed to the aircraft, with the x axis out the nose, the y axis out the right wing and the z axis pointing out the bottom of the aircraft. The aircraft is mathematically represented as a point mass (Assumption 1), which locates all of the aircraft's weight and dynamics at a single point. This removes the impact that moments of inertia have on the dynamics and greatly simplifies the problem. The aircraft's roll angle (ϕ), pitch angle (θ) and yaw angle (ψ) represent the aircraft's orientation in this plane. These angles are collectively referred to as Euler angles. The transformation matrix, shown in Equation 3.1, translates the Euler angles into the actual UAV position in the inertial frame[1].

$$R_b^i = \begin{bmatrix} c(\theta)c(\psi) & -c(\phi)s(\psi) + s(\phi)s(\theta)c(\psi) & s(\phi)s(\psi) + c(\phi)s(\theta)c(\psi) \\ c(\theta)s(\psi) & c(\phi)c(\psi) + s(\phi)s(\theta)s(\psi) & -s(\phi)c(\psi) + c(\phi)s(\theta)s(\psi) \\ -s(\theta) & s(\phi)c(\theta) & c(\phi)c(\theta) \end{bmatrix} \tag{3.1}$$

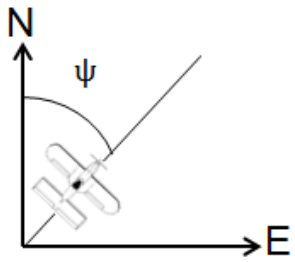
To further simplify the process, the altitude of the UAV is held constant (Assumption 2). To fly at a constant altitude, both in straight and level flight and in a coordinated turn, requires a certain amount of aircraft pitch. The specific pitch angle required is dependent on the aircraft and the flight conditions. Part of the coordinated turn assumption is that the requisite amount of pitch is automatically applied to maintain a constant altitude throughout the turn. The relationship between the pitch and roll angle is shown in Equation 3.2.

$$\theta = 0.1003\,\phi^2 + 0.0592\,\phi + 0.0268 \tag{3.2}$$

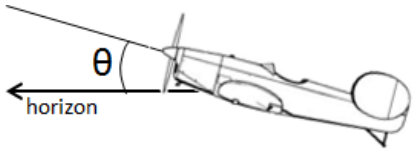
Both angles are defined in radians and Equation 3.2 is determined from flight test data using a tuned Sig Rascal flying at 15m/s. The flight conditions used to determine the relationship between roll and pitch mimic those used during the real-world flight test for the convoy overwatch. Figure 3.1 shows how each of the Euler

[1] c() represents cosine and s() represents sine

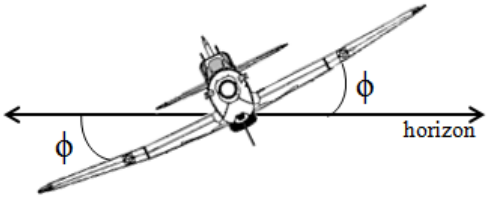
angles are positively defined in the body axis. The yaw angle is defined as the heading angle which defines 0° as North, 90° as East, 180° as South and 270° as West. The pitch angle defines the angle from the horizon to the nose of the aircraft, where a nose up orientation is a positive deflection. The roll angle is positively defined as right wing down and the angle is measured from the horizon to the wing [6].



(a) Positive Yaw Angle



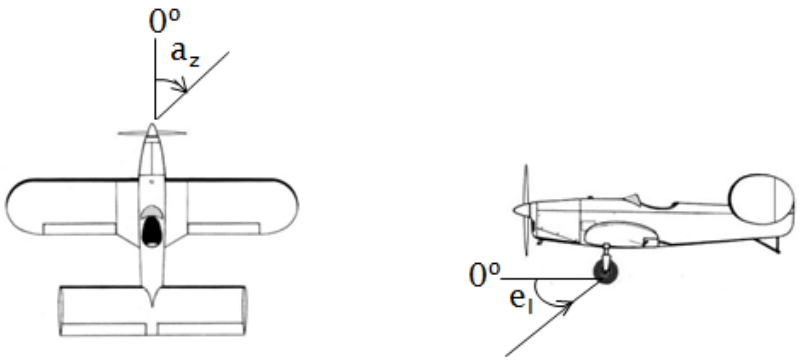
(b) Positive Pitch Angle



(c) Positive Roll Angle

Figure 3.1: Definition of Positive Euler Angles in Body Frame

3.1.3 Gimbal Frame. Lastly, the gimbal frame is fixed to the camera and shows where the camera is pointing relative to the aircraft body. The two angles used to describe the location of the gimbal relative to the body frame are the azimuth angle (a_z) and the elevation angle (e_l). The pictorial definitions of these two angles are shown in Figure 3.2. When the camera is pointed out the nose of the aircraft the azimuth angle is 0° and it increases as the camera swivels clockwise. Likewise, when the camera is pointed at the horizon the elevation angle is 0° and it increases as the camera angles towards the ground until it reaches 90°.



(a) Positive Azimuth Angle Definition (b) Positive Elevation Angle Definition

Figure 3.2: Definition of Azimuth and Elevation Angles in Gimbal Frame

Having the camera oriented directly perpendicular from the body of the aircraft ($e_l = 90°$) is problematic. This orientation represents a singularity in the gimbal control. The gimbal does not know whether to use azimuth or elevation to continue tracking the ground vehicle. This can lead to operator disorientation and visual loss of the vehicle during tracking. This undesirable camera orientation is colloquially referred to as nadir and is one of the challenges facing UAV tracking. Constraints are placed on both the optimal and heuristic-based solutions to prevent the camera from ever being in nadir.

Knowing the camera's aim point, relative to the aircraft's orientation, is crucial for the UAV tracking mission. Since there are two degrees of freedom in the gimbal frame, the transformation matrix (Equation 3.3) includes both a_z and e_l. For any given camera azimuth and elevation, this matrix translates the representing orientation relative to the body frame.

$$R_g^b = \begin{bmatrix} c(e_l)c(a_z) & -s(e_l) & -s(e_l)c(a_z) \\ c(e_l)s(a_z) & c(a_z) & -s(e_l)s(a_z) \\ s(e_l) & 0 & c(e_l) \end{bmatrix} \tag{3.3}$$

3.2 Dynamic Optimization

This section defines the process for developing, testing and implementing the tracking algorithm. An optimization that dynamically considers the vehicle's ground path is required to find the optimal path. The dynamic optimization is divided into the three sections: 1) optimal control, 2) sensor aim point and 3) system constraints. The optimal control section specifically focuses on developing the cost function. The sensor aim point section discusses the equations used to model the gimbal behavior as the UAV tracks the ground vehicle. Lastly, the system constraints ensure the solution is realistic and feasible.

3.2.1 Optimal Control. Finding the optimal control requires a defined set of states and controls, established UAV equations of motion and a well-formulated cost function. There are four states considered in this optimization. The first two states are the UAV's latitude (X_{uav}) and longitude (Y_{uav}). These states are GPS coordinates and they are defined in the inertial frame. The other two states are the roll angle (ϕ) and the yaw angle (ψ), which are two of the Euler angles defined in the body frame. The UAV automatically uses the required pitch control to maintain a constant altitude, this allows the optimizer to neglect the pitch angle (θ) when calculating the optimal path. For the optimization, the location of the ground vehicle is an exogenous input (Assumption 4) and is fed directly into the optimizer.

The aircraft's roll rate ($\dot{\phi}$) is the only control variable used in the optimization. The level turn assumption (Assumption 3) allows for a roll angle to dictate a turn rate. This concept is further explained later in this section, although the important takeaway is that one control variable can sufficiently describe the UAV's lateral-directional motion. Additionally, velocity is not considered as a control variable, but is fixed at the velocity corresponding to the UAV's max L/D. While varying the throttle does increase the UAV's maneuverability, the diminished endurance that results from throttle variation does not justify its use as a control variable. The states and control used in the optimal control problem are summarized in Equation 3.4.

$$\text{States} = \begin{bmatrix} X_{uav} \\ Y_{uav} \\ \phi \\ \psi \end{bmatrix} \qquad \text{Control} = \dot{\phi} \tag{3.4}$$

The solution methodology for solving the dynamic optimization problem is to formulate the continuous problem from the initial time (t_0) to the final time (t_f) as a series of finite pieces. The states and controls are translated into vectors allowing the dynamic optimization problem to be recast as a static optimization. The state vectors are determined using their corresponding equations of motion, which are differential equations that relate the states from one time step to another. The four differential equations of motion that propagate the states forward in time are defined in Equations 3.5 - 3.8.

$$\dot{X}_{uav} = V\cos\psi - V_w\cos\psi_w \tag{3.5}$$

$$\dot{Y}_{uav} = V\sin\psi - V_w\sin\psi_w \tag{3.6}$$

$$\dot{\phi} = u \tag{3.7}$$

$$\dot{\psi} = \frac{g}{V}\tan\phi \tag{3.8}$$

The first order approximation of how the position of the UAV changes with time is defined in Equations 3.5 and 3.6. The UAV's velocity in the inertial frame is a function of its airspeed (V), heading (ψ), wind speed (V_w) and wind direction (ψ_w). It is important to note that while the UAV flies at a near constant airspeed, the presence of wind results in a widely varying ground speed. This distinction is critical because the ground vehicle's speed is constrained to not exceed the ground speed of the UAV (Assumption 6). A scenario that features the vehicle driving faster than the UAV's ground speed presents an unfeasible situation for path planning and therefore must be avoided. The roll rate, shown in Equation 3.7, is the control input that the optimizer uses to produce the optimal path.

The time rate of change of the heading angle, commonly referred to as the turn rate, is shown in Equation 3.8. One of the assumptions made is that the UAV performs coordinated, level turns (Assumption 3). This means that in a turn, the UAV is maintaining a constant altitude, a constant airspeed and is using the correct combination of rudder, aileron and elevator to negate any sideslip. The presence of sideslip during a maneuver adds drag and significantly reduces the UAV's endurance. Additionally, assuming coordinated turns implies that the aircraft is turning for any non-zero bank angle. A positive bank angle corresponds to a right hand turn, while a negative bank angle corresponds to a left hand turn. Relating the turn rate to the UAV's bank angle simplifies the aircraft dynamics to one independent variable. As seen in Equation 3.8, the turn rate of the UAV is only dependent on the bank angle (ϕ) of the UAV because the airspeed is held constant.

By defining the states and control and determining the equations of motion, there is sufficient information to construct the cost function. Shown in Equation 3.9, the cost function aims to minimize the weighted sum of the control and slant range (SR) error. The cost function represents the desire to keep the UAV a certain distance from the ground vehicle while using the minimum required control. In Equation 3.9, the first term penalizes deviation from desired slant range and the second term penalizes the control. Both the slant range and control terms are normalized relative to constant values so that the two terms can be equally weighted relative to each other.

$$J = \int_{t_0}^{t_f} \left[\alpha \left(\frac{SR - SR_{desired}}{SR_{desired}} \right)^2 + (1 - \alpha) \left(\frac{u}{u_{\max}} \right)^2 \right] dt \tag{3.9}$$

The cost function can be further conditioned to favor either reducing control effort or deviation from the desired slant range by varying the weight factor (α). When $\alpha = 0.5$, the two terms are equally weighted according to their normalized values and the optimizer seeks to decrease both terms evenly. When $\alpha > 0.5$, the deviation from the desired slant range receives greater emphasis, while $\alpha < 0.5$ corresponds with emphasizing the reduction of control effort. Regardless of the value chosen for

α, the resulting solution defines the optimal path pertaining to that specific weight. This creates an infinite number of optimal solutions, also called a Pareto front [13]. Pareto fronts occur in all multi-objective cost functions because of the variable weight factors attached to each term. In essence, a new optimization problem must be performed to differentiate between the infinite number of optimal solutions. The system requirements and UAV performance drive the decision making process to determine the appropriate α for the problem. Chapter IV analyzes how changing α affects both slant range and control effort to determine the appropriate value to use for generating the optimal paths.

The slant range is defined in Equation 3.10 and represents the three dimensional distance between the UAV and the ground vehicle. Due to the constant altitude assumption (Assumption 2), the slant range and the two dimensional standoff distance are directly related. For this reason, the two words are used synonymously to represent a fixed distance that the SUAS tries to maintain from the moving vehicle.

$$SR = \sqrt{(X_{gv} - X_{uav})^2 + (Y_{gv} - Y_{uav})^2 + h^2} \tag{3.10}$$

Slant range is considered in the cost function because it represents the true distance between the UAV and the ground vehicle. The pixel density and image FOV determine the convoy commander's situational awareness (SA). If the UAV is too close, there is a reduction in the FOV due to the UAV's proximity to the convoy. Conversely, if the UAV is too far away, the pixel density is not sufficient to produce actionable intelligence. Either way, the commander loses SA and subsequently the effectiveness that an autonomous UAV can provide. The goal is to find the point where the UAV should operate to improve SA, while meeting the operationally dependent constraints.

The control is included in the cost function for three reasons. The first one is constraining control effort directly correlates to increasing endurance. Specifically for the Sig Rascal, no quantitative evidence relating roll rate to power consumed

is available; however, the general principle makes intuitive sense. Secondly, if the control is not included in the cost function, the optimizer will use the full extent of the control regime to tightly maintain the desired slant range. This typically results in an undesirable maneuvers because the optimal solution is a series of bang-bang control inputs. Lastly, reducing the roll rate of the aircraft improves the image quality of the video. If the SUAS is constantly maneuvering, the sporadic motion may result in blurry and potentially unusable footage, which defeats the entire purpose of the surveillance mission. For all these reasons, it is important to consider control effort in the cost function formulation.

The cost function, in Equation 3.9, is used to determine the optimal path, but it can also be thought of as a SA metric. The multi-objective function leverages pixel density against image blurriness by constantly seeking the solution that maintains a minimal amount of control effort and slant range error. In theory, the optimal path not only increases aircraft endurance while adhering to a desire standoff, it also gives the user consistent video imagery. This concept of optimality is multi-faceted and can be tailored to fit the UAV's performance and surveillance requirements.

Defining the states and controls, determining the equations of motion and building the cost function are critical keys to performing the dynamic optimization. The next step is to find the optimal path for a given ground profile by using nonlinear programming (NLP) to solve the optimal control problem. The `Matlab`® function *fmincon* provides the computational environment to evaluate the cost function shown in Equation 3.9. The Interior Point Method (IPM) is the solver used by *fmincon* that takes the initial guess for the optimal control and converges on the control vector that minimizes the cost. This method satisfies the KKT conditions by using a successive series of descent steps [15]. The IPM is used as the optimal control solver because it scales well to solve complex problems and provides a relatively fast, reliable and robust solution method.

3.2.2 Sensor Aim Point. The camera orientation onboard the UAV is of utmost importance. The entire purpose of video surveillance is defeated if the camera does not point in the right direction. One of the assumptions made is the camera always points at the target (Assumption 5). This implies that the location of the vehicle is known at each moment in time and the camera's dynamics are significantly faster than the vehicle (Assumption 4). Both of these assumptions are reasonable and are essential for determining the a_z and e_l required to keep the camera pointing at the ground vehicle. The camera's aim point in the inertial frame is defined in Equation 3.11 and is relative to the gimbal and aircraft's orientation. R_b^i and R_g^b represent the coordinate transformation matrices defined in Equations 3.1 and 3.3 respectively. The camera's mounted location with respect to the gimbal frame is defined in Equation 3.12. Here, the value of $\vec{s}$ signifies that the neutral camera position is oriented in the positive x direction (out the nose of the aircraft).

$$\vec{\text{aim}} = R_b^i R_g^b \vec{s} \tag{3.11}$$

$$\vec{s} = [1, 0, 0]^T \tag{3.12}$$

The camera's aim point is known because the location of the UAV and the ground vehicle are known. The distance between the UAV and the ground vehicle is found by differencing the respective latitude, longitude and altitude of the UAV and the ground vehicle (Equation 3.13). The aim vector is the unit normal vector of Equation 3.13 and it represents the pointing orientation of the camera in the inertial frame.

$$\vec{\text{dist}} = \begin{bmatrix} X_{gv} - X_{uav} \\ Y_{gv} - Y_{uav} \\ h \end{bmatrix} \tag{3.13}$$

$$\vec{\text{aim}} = \frac{\vec{\text{dist}}}{\|\vec{\text{dist}}\|} \tag{3.14}$$

Rearranging Equation 3.14 results in Equation 3.15.

$$R_g^b \vec{s} = (R_b^i)^T \vec{\text{aim}} \tag{3.15}$$

The system of equations created from plugging all the requisite values into Equation 3.15 are solved to produce the corresponding a_z and e_l angles for any given ϕ, θ, ψ and $\vec{aim}$. The azimuth and elevation angles are found in Equations 3.16 and 3.17 and can be used to track the orientation of the camera throughout the flight.

$$a_z = \tan^{-1}\left(\frac{[-c(\phi)s(\psi) + s(\phi)s(\theta)c(\psi)]\text{aim}_x + [c(\phi)c(\psi) + s(\phi)s(\theta)s(\psi)]\text{aim}_y + s(\phi)c(\theta)\text{aim}_z}{c(\theta)c(\psi)\text{aim}_x + c(\theta)s(\psi)\text{aim}_y - s(\theta)\text{aim}_z}\right) \tag{3.16}$$

$$e_l = \sin^{-1}([s(\phi)s(\psi)+c(\phi)s(\theta)c(\psi)]\text{aim}_x - [s(\phi)c(\psi)-c(\phi)s(\theta)s(\psi)]\text{aim}_y + c(\phi)c(\theta)\text{aim}_z) \tag{3.17}$$

The azimuth angle is computed using the four quadrant, inverse tangent function which allows it to range from 0° to 360°. Mathematically, the elevation angle can range from −90° to 90°, although a negative elevation angle means the camera is pointed up at the UAV. Both Equation 3.16 and 3.17 represent the valid angles that the gimbaled camera can achieve in the correct coordinate frame.

3.2.3 System Constraints. For any dynamic optimization problem to be representative, the mathematical formulation must include the system constraints. These constraints can be on both the states and controls and prevent the optimizer from finding a solution in a physically infeasible space. The equations of motion, defined in Equations 3.5 - 3.8, are called dynamic constraints and act as equality constraints, forcing the optimizer to satisfy each state equation at each time step.

In the context of this optimization, several path constraints are placed on the system to model the real-world dynamics of the Sig Rascal. Table 3.1 highlights the path constraints for the elevation angle (e_l), roll angle (ϕ) and roll rate ($\dot{\phi}$). The elevation constraints are chosen to prevent the optimizer from finding a solution that

allows the camera to nadir ($e_l = 90°$) or is outside its physical operating range. The bank angle is constrained to match the actual minimum turn radius of the Sig Rascal. Remember that a bank angle corresponds with a turn rate (Equation 3.8) and therefore must be constrained even if the aircraft is capable of achieving a higher bank angle. Lastly, the maximum and minimum roll rates were determined based on real-world telemetry data. The roll rate constraint is the least likely to be active because of its inclusion in the cost function. Therefore the limits set for this value are not nearly as important as they are for the other two values.

Table 3.1: Path Constraints for Dynamic Optimization

Variable	**Min**	**Max**
e_l	$0°$	$80°$
ϕ	$-40°$	$40°$
$\dot{\phi}$	-100 deg/s	100 deg/s

3.3 Flight Test

Due to hardware and computational limitations, `Matlab`® cannot be implemented onboard the APM. Therefore, empirical methods are required to develop a model for approximating the optimal solution. The goal of the flight test is to develop a heuristic algorithm that best approximates the optimal solution for that corresponding path and runs onboard the autopilot. Captain Charles Neal [17] worked in conjunction with this thesis effort to develop and implement the heuristic-based approximation of an optimal path. He was specifically responsible for altering the autopilot code, running the majority of the HIL simulations and performing a design of experiments (DOE) to determine the best parameter settings for the final heuristic-based solution.

3.3.1 Heuristic-Based Approach. The fundamental idea of the heuristic-based approach is manipulating the autopilot's loiter logic to allow for the UAV to

loiter about the moving ground vehicle. When the ground vehicle is stationary, the UAV can loiter at the desired slant range with minimal control effort. The difficulty arises when the ground vehicle begins to move. Using the same strategy, the UAV attempts to loiter about the now moving ground vehicle. The loiter radius, loiter range and the lead time are the three parameters chosen to condition the autopilot's behavior to approximate the optimal path. Figure 3.3 displays how the loiter radius and loiter range are defined relative to the point of interest.

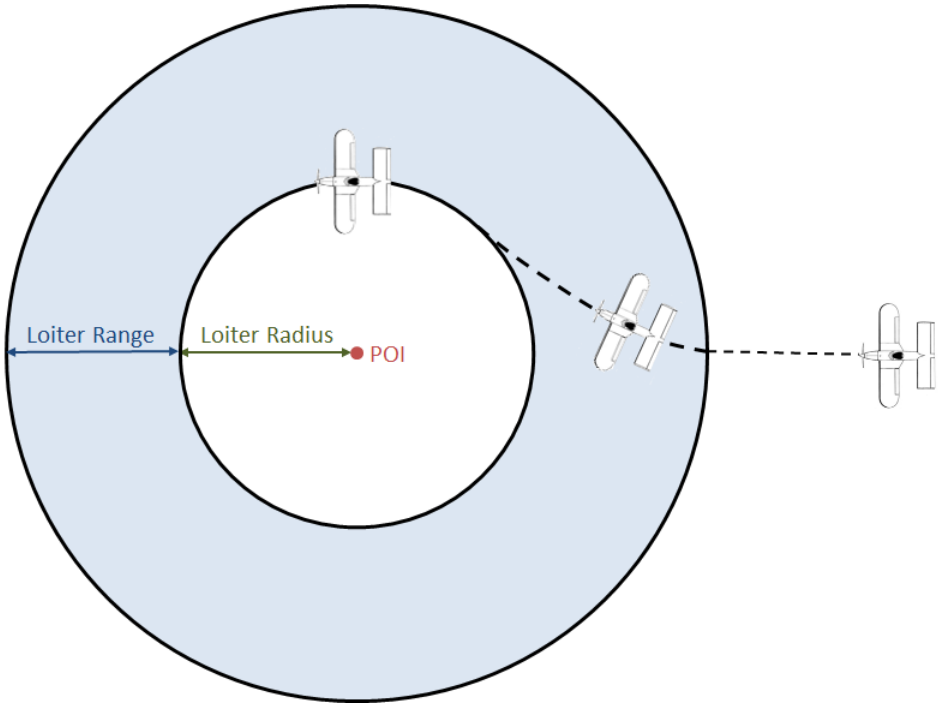


Figure 3.3: Definition of Loiter Radius and Loiter Range

The loiter radius is the same as the standoff distance and is found by differencing the latitude and longitude of the UAV and the ground vehicle. Given the constant UAV altitude, a loiter radius directly corresponds to a slant range. For any desired slant range and altitude, the corresponding loiter radius can be determined and plugged into the autopilot. The purpose for using the loiter radius as a parameter is to see if varying the loiter radius affects how the heuristic-based method approximates the optimal path. This can be directly measured by how closely the UAV adheres to the desired slant range while following a moving target.

Secondly, the loiter range is evaluated to see its impacts on the UAV's flight path. The loiter range dictates how abruptly the UAV maneuvers to maintain a loiter. Shown in Figure 3.3, the loiter range is a distance that defines a region outside of the desired loiter radius. When the UAV is outside this region, it flies directly at the point of interest. Once the UAV crosses into the region defined by the loiter range, the autopilot maneuvers to tangentially intersect the loiter radius. When the POI is moving, the size of the loiter range dictates how closely the UAV adheres to the desired loiter radius. A larger loiter range allows the UAV to perform gentler maneuvers as it flies to intersect the desired loiter radius. Conversely, a smaller loiter range keeps the UAV tightly bound to the desired loiter radius.

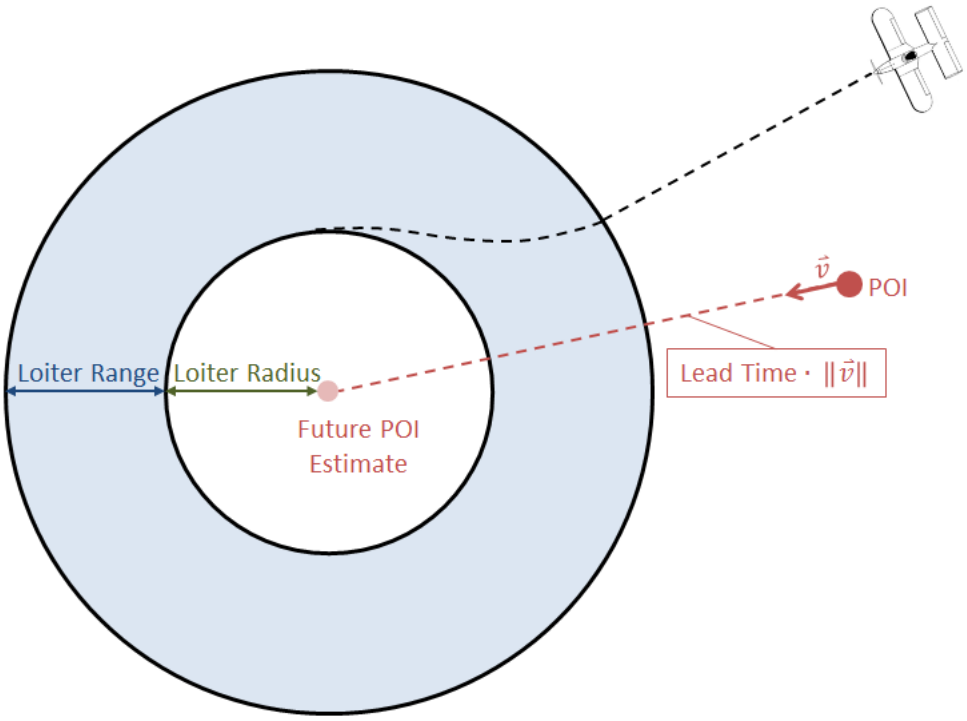


Figure 3.4: Definition of Loiter Radius, Loiter Range and Lead Time

The autopilot's future path estimation technique is portrayed in Figure 3.4. The lead time represents a time constant that is used to predict the POI's future location and is the last parameter used in the heuristic-based method. The autopilot first determines the ground vehicle's speed and direction. The future POI location

is found by multiplying the vehicle's current speed by the lead time and adding that distance in the direction that the ground vehicle travels. The loiter logic then adjusts how the UAV behaves based on UAV's predicted future location, while still pointing the camera at the ground vehicle's current location. As shown in Figure 3.4, the UAV loiters about the predicted POI location not the actual POI itself. As the POI moves, the estimated location moves accordingly. This prediction is a simple linear extrapolation of the POI's velocity over a given time and is updated every second. The lead time is manipulated to see how this linear prediction of future location affects the actual behavior versus the optimal.

All three of these parameters are used as variables to evaluate the UAV's actual flight path relative to the optimal path. The multi-objective cost function for generating the optimal path considers two terms, slant range and control. The intent of the heuristic-based method is to approximate the optimal path, by simultaneously reducing the UAV's control effort and slant range error. The loiter radius, range and lead time are purposefully chosen to condition the autopilot to make decisions similar to the optimal solution.

As previously mentioned, the loiter radius and the desired slant range are directly related. For any given loiter radius, the autopilot has a built in functionality that maneuvers the UAV to keep that desired radius. The user simply needs the desired altitude and slant range to find the corresponding loiter radius. Increasing the loiter range implicitly reduces the control effort, because as the loiter range increases, the UAV makes gentler turns that result in smaller control outputs. Although lead time is not explicitly factored into the cost function, the optimal path relies on future path knowledge to make decisions. The cost function integrates all of the slant ranges and control inputs over a fixed time to determine the optimal path. To accomplish this, the optimizer must know exactly where the UAV and ground vehicle will be over that time interval. This notion of future path knowledge is further explained in Chapter IV. The important link is that the heuristic-based method uses the lead

time to estimate the future location of the ground vehicle to better approximate the optimal path.

While these three parameters represent specific terms in the cost function, they are by no means mutually exclusive. Changing the loiter range does not simply affect the control effort but also impacts how the UAV adheres to the desired slant range. Likewise, lead time can affect both the UAVs control effort and slant range error. For this reason, flight test is required to determine the appropriate settings for each parameter to best approximate the optimal path.

3.3.2 Flight Test Plan. A series of flight tests were completed to investigate how to best approximate the optimal path for a given ground path. The process of approximating an optimal path with heuristic-based methods was iterative and required multiple test events. The goal of the first flight test was to accomplish the autonomous ground vehicle tracking mission. Autonomous tracking was accomplished by constantly locating the UAV's loiter point on the current coordinates of the ground vehicle. This strategy formed the basis for the eventual heuristic-based approximation. The initial flight test was important because it provided the foundation for building the heuristic-based approximation.

Once the autonomous tracking capability was demonstrated during the first flight test, the three parameters of loiter radius, loiter range and lead time were incorporated to test their impact. Neal [17] performed a design of experiments (DOE) to determine what values for the three parameters most closely approximated the optimal path. Refer to [17] for a more in-depth analysis of the process that determined how many tests were performed, what range of values were used and how the three parameters were paired together. Table 3.2 shows the first stage of the intermediary flight test matrix flown in HIL to evaluate the impact of these parameters and their covariance relative to each other. HIL simulations were used to execute the DOE because of the numerous flight tests required for statistically relevant results coupled with the limited flight test time.

Table 3.2: First Stage Experimental Design [17]

Test	**Loiter Range (m)**	**Loiter Radius (m)**	**Lead Time (sec)**
1	182	183	8.9
2	40	125	5.0
3	58	67	8.9
4	120	125	0
5	120	125	5.0
6	58	183	8.90
7	120	125	5.0
8	200	125	5.0
9	182	67	8.9
10	182	183	1.1
11	182	67	1.1
12	120	125	10.0
13	58	183	1.1
14	120	50	5.0
15	120	200	5.0
16	58	67	1.1

The results of this DOE were evaluated and used to create a second stage experiment that narrowed the scope of the parameters, shown in Table 3.3.

Table 3.3: Second Stage Experimental Design [17]

Test	**Loiter Range (m)**	**Loiter Radius (m)**
1	40	152
2	20	150
3	62	130
4	40	130
5	40	130
6	20	110
7	60	110
8	18	130
9	40	108
10	60	150

The data gathered from these intermediary flight tests was used to determine the parameter values that adhere best to the optimal path. A final flight test was used to evaluate the optimality of parameter values chosen from the DOE. The results of this final flight test were compared with its corresponding optimal path to determine the significance the different parameters have on approximating the optimal path.

The UAV's altitude, standoff distance and velocity selected for the optimization and subsequent flight test are chosen to comply with AFIT's test safety constraints and are not necessarily operationally relevant. The purpose of the flight test is to prove that the UAV can heuristically approximate the optimal solution regardless of the desired slant range, which can be change due to the onboard camera, convoy size as well as numerous other reasons.

The methodology presented in this chapter defined the coordinate frames used, created the framework for the dynamic optimization algorithm and created the process for finding a heuristic-based solution. The cost function that dictates the optimal

solution was determined based on the framework established from Chapter I. Additionally, the heuristic-based approximation relied on the three parameters of loiter radius, loiter range and lead time. The process for using these values was briefly discussed but is more deeply considered in the work of Neal [17].

Creating an optimal path for a given ground path and approximating that process onboard a real-world UAV were the two objectives stated in Chapter I. Chapter IV accomplishes the first goal by using the data from this chapter to determine the optimal paths. Subsequently, Chapter V achieves the second goal of incorporating a heuristic-based tracking algorithm onboard the UAV. Both of these chapters rely on the methodology in this chapter and build upon the foundation laid by Chapters I-III.

IV. Optimal Path Solutions

This chapter accomplishes the first goal of this thesis by developing two optimal UAV flight path algorithms for a given ground vehicle path. The first section of this chapter shows the specific equations used in both of the optimal path algorithms. Here, the equations of motion, cost function and system constraints, discussed in Chapter III, are discretized over fixed time intervals. The two algorithms used to determine the optimal solution are outlined in the subsequent sections. The first algorithm uses the full ground vehicle path and plans the optimal route knowing the exact ground vehicle location at every time step. This a priori knowledge approach is unrealistic and therefore impossible to implement in the real world, but it is the "best" of all the possible paths. The second algorithm assumes that the UAV can project the ground vehicle's path ahead a short time into the future and plans the optimal path based on the limited future knowledge of the vehicle's path. Essentially, this method is a limited version of the a priori approach and it represents a more feasible approach for implementation in the real world. Studying how different "look ahead" times affect the optimal solution will aid in algorithm selection for eventual implementation.

4.1 Optimal Path Algorithm Equations

The first step in calculating the optimal path is to determine the initial conditions. The starting location and orientation of the UAV is necessary to obtain the optimal control solution. Equation 4.1 shows the four values required to start the process. Additionally, the ground vehicle's path (X_{gv} and Y_{gv}), the UAV's airspeed (V) and the wind speed and direction (V_w and ψ_w) are treated as fixed parameters.

$$\begin{aligned} X_{uav}(0) &= X_{uav_o} \\ Y_{uav}(0) &= Y_{uav_o} \\ \phi(0) &= \phi_o \\ \psi(0) &= \psi_o \end{aligned} \tag{4.1}$$

Along with the initial conditions, an initial guess of the control trajectory is required. The control variable is determined at each time step, which minimizes the cost, as calculated by the cost function in Equation 4.7.

The equations of motion, defined in continuous time, in Equations 3.5 - 3.8 are discretized, based on a fixed Δt, in Equations 4.2 - 4.5. For the fixed time step, Δt, these equations are used as a first order approximation to find the states at the next step. Equation 4.2 uses the control and the current roll angle to determine the roll angle of the next time step.

$$\phi(i+1) = \phi(i) + u(i)\,\Delta t \tag{4.2}$$

The heading angle at the future time step is found using Equation 4.3 and is dependent on the roll angle at the current time.

$$\psi(i+1) = \psi(i) + \left[\frac{g}{V(i)}\tan\phi(i)\right]\Delta t \tag{4.3}$$

The location of the UAV at the proceeding time step is given as Equations 4.4 and 4.5.

$$X_{uav}(i+1) = X_{uav}(i) + [V\cos\psi(i) - V_w(i)\cos\psi_w(i)]\,\Delta t \tag{4.4}$$

$$Y_{uav}(i+1) = Y_{uav}(i) + [V\sin\psi(i) - V_w(i)\sin\psi_w(i)]\,\Delta t \tag{4.5}$$

The values for V, V_w and ψ_w can either be constant or time varying. For initial simulations, these values are set at constant values. However, Chapter V compares the simulation results to the flight test data and, in that case, uses the identical airspeed, wind speed and wind direction that were measured by the real-world UAV in flight test. Incorporating the real-world, time varying data in the algorithm is beneficial because it allows for a more realistic comparison.

The locations of the UAV and the ground vehicle are required for calculating the slant range at that particular interval (Equation 4.6).

$$SR(i) = \sqrt{(X_{gv}(i) - X_{uav}(i))^2 + (Y_{gv}(i) - Y_{uav}(i))^2 + h^2} \tag{4.6}$$

The cost functional, shown in Equation 4.7, is the summation of the cost at each step. The optimal control is characterized by the control vector (u) that minimizes the scalar value J in Equation 4.7. This optimal control vector is then used to determine the optimal path.

$$J = \sum_{i=0}^{N} \left[\alpha \left(\frac{SR(i) - SR_{desired}}{SR_{desired}} \right)^2 + (1 - \alpha) \left(\frac{u(i)}{u_{\max}} \right)^2 \right] \Delta t \tag{4.7}$$

4.2 Full Path Algorithm

The full path algorithm relies on the complete future knowledge of the vehicle's location to create the optimal path. Actual GPS coordinates from the flight test telemetry data are gathered by driving the ground vehicle in the "figure 8" path, shown in Figure 2.6. These coordinates are then fed into the optimizer and represent the ground vehicle's location at all times. Even though the road network used for each test is the same, the vehicle can drive that same path at various speeds, resulting in different UAV optimal paths. For this reason it is difficult to compare one optimal path to another because of the variability of the ground profiles. The optimal path solution minimizes the weighted sum of the control effort and the slant range error between the UAV and the ground vehicle.

The algorithm, outlined in Algorithm 1, is used for determining the optimal path of the UAV. The value of N represents the total number of discretized nodes (points in time) that span the entirety of the ground vehicle's path. The solution strategy assumes a constant frequency of nodes placed throughout time and therefore the value of N equals the product of the mesh frequency ($Freq$) and the final time (t_f). Next, the initial conditions must be determined, the wind and airspeed parameters acquired

and an initial guess of the control vector made. Using the initial control vector guess, the discretized state equations (Equations 4.2 - 4.5) are propagated forward to the final time (t_f). This process successfully recasts the dynamic optimization problem of UAV convoy overwatch into a static optimization. The state and control vectors (both length N), are fed into the optimizer. The optimizer converges on the control and state vectors that minimize the scalar cost value (Equation 4.7) and satisfies the dynamic and path constraints. The optimal states are returned with the optimal control once all of the convergence criteria are satisfied.

Algorithm 1 : Full Path Algorithm

Determine total number of nodes: $N = t_f \times Freq$

Set initial conditions

Acquire ground vehicle path and wind information (used as fixed parameters)

Guess control vector (u)

Propagate state equations forward

Optimizer:

Find u which minimizes cost function

Satisfy dynamic constraints

Satisfy path constraints

Result: Optimal control and states found for $t_0 : \Delta t : t_f$ for ground vehicle path

In this solution methodology, all ground vehicle turns and speed changes are known and are considered in the optimal path solution. The benefit of knowing the future locations of the ground vehicle is that the optimal solution accounts for all vehicle behavior. This foreknowledge means that the full path approach should achieve a lower cost value than any look ahead strategy. However, even though the future ground vehicle path is known a priori, it does not guarantee that the full path method will converge on the global minimum. Optimizer tolerances, initial control guesses and different ground profiles can all contribute to the optimizer converging on a local, rather than global, minimum. This is not necessarily bad because the

local minimum could be just slightly higher than the global minimum, resulting in a nearly identical optimal path or a different path with a similar cost. The uncertainty of achieving a global minimum allows for the unlikely possibility that another method might result in a lower cost function value than the full path method. Conceptually, any other approach or heuristic-based approximation can at best achieve the same results, but will most likely yield a solution with a higher cost. A mesh refinement (changing the choice of Δt) and weight factor (α) analysis is performed to help increase the confidence in the obtained optimal path.

4.2.1 Mesh Refinement. The optimal solution is calculated over a uniformly spaced, finite number of nodes. These nodes create a mesh over the entire interval and the mesh frequency affects both the accuracy and timeliness of the solution. The goal of the mesh refinement study is to determine the smallest mesh frequency required to achieve the optimal path without overburdening the optimizer with needless computations. To determine the appropriate frequency, the same ground path is used for a variety of mesh frequencies. The coordinates where the paths converge, regardless of frequency, indicates the desired optimal path. Figure 4.1 shows a plot of four different optimal paths calculated on meshes with frequencies of 1, 1.5, 2 and 3 Hz. Time is not explicitly labeled in Figure 4.1 because it shows the path profiles. Every 60 seconds, an aircraft is plotted for each optimal path and a square is plotted for the ground vehicle to give a visual reference of the UAV's location relative to the ground vehicle. Table 4.1 displays the optimization results, shown in Figure 4.1, to include the number of nodes, cost and solution time for each of the mesh frequencies [1].

[1] All solutions computed on a Samsung ATIV Smart PC pro (4GB of RAM) using `Matlab`® 2012b under default settings

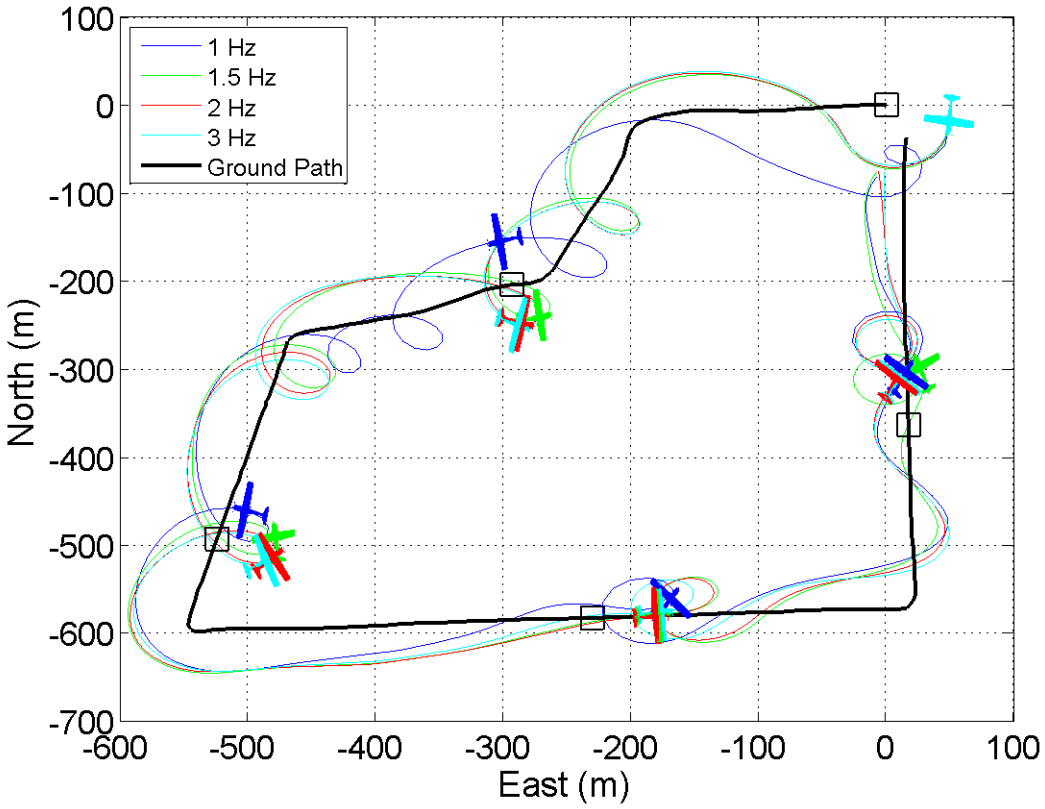


Figure 4.1: Optimal Paths for Multiple Mesh Refinements

Table 4.1: Optimization Results for Figure 4.1

Mesh Freq (Hz)	**# of Nodes**	**Cost (J)**	**Run Time (min)**
1	279	1.41	5.14
1.5	419	0.23	28.9
2	558	0.21	80.9
3	837	0.20	245.5

In Figure 4.1, there is a noticeable difference between the optimal path for the 1 Hz mesh and the other three. The disparity between the cost functions for these two groups is the largest determining factor for finding the best mesh frequency. Shown in Table 4.1, the 1.5, 2 and 3 Hz paths all have cost function values around 0.2, while the 1 Hz path is almost seven times higher. While the costs of the 1.5, 2 and 3 Hz paths are nearly the same, the time required to converge on the solution significantly increases with a higher frequency. The 52 minute jump in run time from the 1.5 Hz to the 2 Hz solution only yields a 0.02 reduction in the cost. The 1.5 Hz solution's close proximity to the higher frequency solutions and more efficient run time makes it the lowest possible frequency that still results in the optimal path. Even though the higher mesh frequencies result in slightly more optimal paths, the relative benefit does not outweigh the relative cost of computer efficiency and time. For this reason, all remaining simulations use 1.5 Hz as the mesh frequency.

Choosing the mesh frequency of 1.5 Hz only applies for this general path and the corresponding ground vehicle and UAV velocities. Any significant changes to the path or the vehicles involved could cause a shift in the required mesh density. Just as the 1 Hz mesh is insufficient for plotting the optimal path for this particular ground profile, 1.5 Hz may prove insufficient for a different profile. Using a rigid node spacing mesh means that the nodes must be close enough to account for the high gradient regions. Path variability is part of the reason the figure 8 path is chosen for both simulation and flight test. The figure 8 ground profile has multiple turns of varying directions and

degrees, which increases the complexity of the ground vehicle path and subsequently increases the difficulty of converging to the optimal path. Since a mesh frequency of 1.5 Hz adequately covers this ground vehicle path profile, it should also apply to more subdued and predictable ground profiles as well. Regardless, it is important to realize the impact that mesh frequency has on the optimal path calculation.

4.2.2 Cost Function Weight Decision. The optimal path is determined by minimizing the weighted sum of the slant range error and control effort in a multi-objective cost function (Equation 4.7). Due to the multiple terms, the cost function can be further conditioned to more accurately represent the user requirements. The variable α is used to adjust the weight the optimizer puts on each term. A greater weight value corresponds to an increased emphasis within the optimal solution. The goal is to determine the appropriate value of α for generating the UAV convoy over-watch paths. Remember that each solution, regardless of α, represents an optimal solution. In the context of the tracking problem, there exists a value that gives the user the best combination of the two variables of slant range and roll rate. The value of α is varied from 0.1 to 0.95 to create the Pareto front. An optimization of all the optimal solutions is performed to find which solution along the Pareto front yields the best path.

The two end values of 0 and 1 are not considered because at those α values the cost function fundamentally changes. By choosing either 0 or 1, either the slant range or control effort is no longer considered in the process. For $\alpha = 0$, the slant range is neglected and the resulting optimal path is simply a straight line with no control inputs. On the other extreme, $\alpha = 1$ signifies that the UAV can use the entire airplane control range to closely maintain the slant range. While this may sound desirable, ignoring control in the cost function allows for extreme control inputs resulting in undesirable flight paths. Therefore, the desired answer must lie in between these two extremes.

Figure 4.2 shows the impacts that varying α has on the roll rate and slant range. Each optimal solution has the same desired standoff distance from the ground vehicle (150 m) and identical initial conditions. The plots in the first column show how the slant range error and roll rate vary with the value of α. The plots in the second column show their corresponding standard deviations. The values of each optimal solution are not as important as the trends that result when α varies.

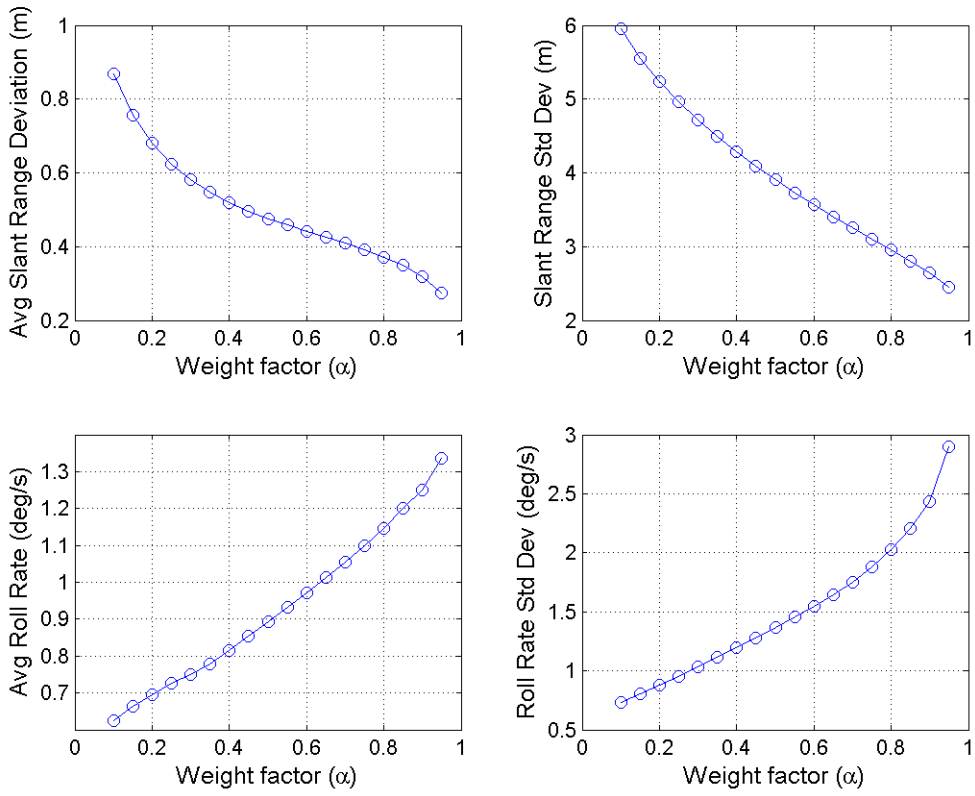


Figure 4.2: Average and Standard Deviation of Slant Range and Roll Rate (150 m Desired Standoff)

Notice in Figure 4.2, as α increases the emphasis of the optimizer transitions from favoring roll rate to slant range. This trend is shown in both the decrease in slant range error and standard deviation as well as the increase in the roll rate and standard deviation. This graph makes intuitive sense and confirms the conceptual understanding of how the weight factors affect the optimal solution. The challenge is

selecting the desired value of α to use for all subsequent paths. This decision is made by evaluating the relative effects that α has on both slant range and control effort. There is a direct correlation between slant range error in the optimizer and real-world performance. The comparison between control effort in the simulation and endurance of the aircraft is not as straightforward. Unfortunately, no direct correlation between roll rate and overall aircraft endurance is determined in this research. This does not discount the rationale for placing the optimizers' control variable in the cost function, but it does make it difficult to make definitive decisions based on small variations in control. Only a 0.65 deg/s difference exists between the average roll rates from $\alpha = 0.1$ to $\alpha = 0.95$. Making a decision within that small of a range, for something that cannot be definitively correlated is unwise. Therefore, $\alpha = 0.95$ is chosen as the weight factor. At this α, there is a marginal increase in control effort and a large decrease in the variability of the slant range error.

The results from Figure 4.2 show the relationship between the two objectives and the weight factor. To verify that these results are valid, an α sweep is performed for a different ground path with a different set of initial conditions. Running the new α sweep reveals an interesting anomaly that yields non-intuitive results. Shown in Figure 4.3, a significant increase in both slant range error and roll rate occur as α increases from 0.3 to 0.35. According to this graph, it appears that some optimal solutions are better than others. The graph in Figure 4.3 shows that increasing the weight to reduce the slant range error actually increases the slant range error. This illogical result highlights the inner complexities of the optimization process. Another inconsistency occurs as α increases from 0.8 to 0.85. The noticeable dip in the roll rate suggests that some phenomenon occurs at that location to cause the deviation.

These irregularities highlight the existence of bifurcation points in the solution. A bifurcation point can simplistically be thought of as a "fork in the road". Within the available search space, different values of α in the cost function can cause the optimizer to converge to distinctly different paths. As α changes, the priority between minimizing control and slant range error in the cost function changes. In certain cases,

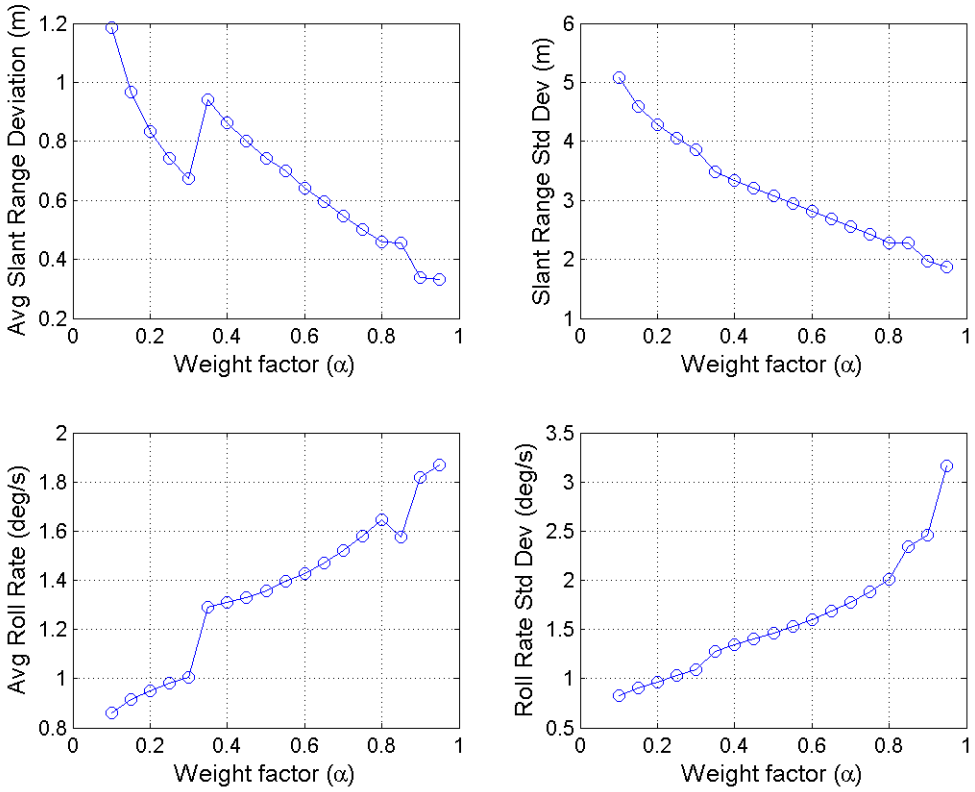


Figure 4.3: Average and Standard Deviation of Slant Range and Roll Rate (100 m Desired Standoff)

there exist bifurcation points that highlight the effect that various values of α have on the optimal path. The impact of the bifurcation points is seen in Figure 4.4.

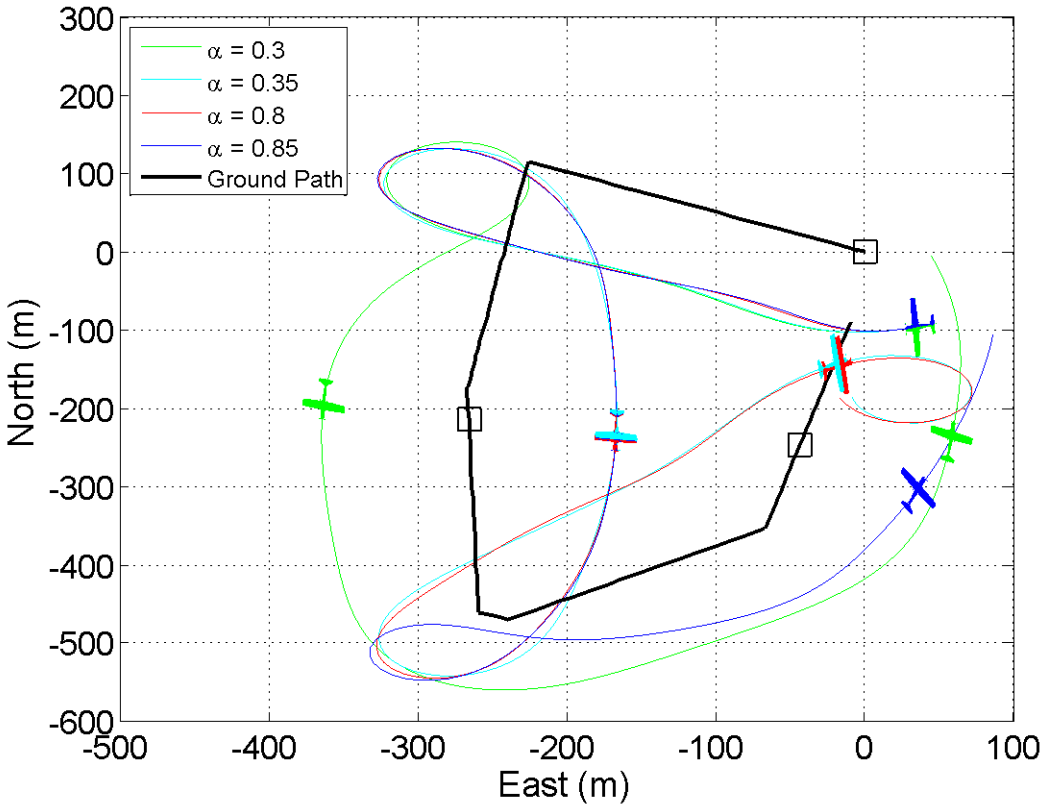


Figure 4.4: UAV Path Comparison: $\alpha = 0.3$, $\alpha = 0.35$, $\alpha = 0.8$ and $\alpha = 0.85$

Figure 4.4 plots the optimal paths of a UAV that has a 100 m desired standoff and has weight factors of $\alpha = 0.3$, $\alpha = 0.35$, $\alpha = 0.8$ and $\alpha = 0.85$. The ground vehicle drives the bottom half of the figure 8 path for a shorter, uncluttered path profile. The shortened path is simulated to plainly show the difference between the optimal paths for their respective α values. Initially, all four paths are nearly identical. The first bifurcation point occurs as the ground vehicle makes the first turn. When accounting for the initial ground vehicle's turn, the UAV has two possible solutions. One reduces the control effort by continuing to turn around and fly to the outside of the ground vehicle's profile, while the second uses control to roll out of the turn to better reduce the slant range error. The transition from $\alpha = 0.3$ to $\alpha = 0.35$

tips the balance from continuing the turn, to rolling out to achieve a better slant range. Counterintuitively, the optimizer's result to maneuver for an improved slant range ends up resulting in a higher average slant range error over the whole path. This occurs because this bifurcation point leads to two distinct local minimums and convergence on the respective local minimum is contingent on the solution obtained by the optimizer at the ground vehicle's first turn.

As α increases from 0.35 to 0.8, the resulting optimal paths do not change. The nearly identical paths show that changing α within this region does not force a tradeoff between slant range error and control effort. Another bifurcation point occurs between $\alpha = 0.8$ and $\alpha = 0.85$. At high α values, the cost function is weighted to favor reducing slant range error over control effort. Therefore, the optimal path at $\alpha = 0.85$ features a more aggressive UAV maneuver that further decreases the slant range error. This bifurcation point coincides with the ground vehicle's second sharp turn. For this specific ground path, both of the bifurcation points are induced when the ground vehicle drastically alters its direction. Substantial changes in ground vehicle direction result in scenarios where the minimization of the UAV's control effort and slant range error are conflicting. When the tradeoff between the two objectives is significant enough, a bifurcation point forms. This specific ground profile, coupled with the UAV's initial conditions yields two bifurcation points, which translates into three possible local minimum optimal path solutions.

The existence of bifurcation points presents potential pitfalls for generating a consistent optimal path. The optimal path is a function of both the ground vehicle and the UAV as well as the optimizer settings. This means different desired standoff distances, initial conditions, wind conditions and convoy behavior can result in vastly different optimal paths with bifurcation points at multiple locations. Using $\alpha = 0.95$ as the weight factor still makes sense despite the existence of the bifurcation points. The marginal increase in roll rate is acceptable when considering the increased tracking performance. From a pure cost function perspective, the cost for both Figures 4.2 and 4.3 is lowest at $\alpha = 0.95$. Bifurcation points are an interesting by-product

of dynamic optimization and factor into the reasoning for selecting $\alpha = 0.95$ as the weight factor for the cost function.

4.3 Look Ahead Algorithm

In the look ahead algorithm, only a few seconds of the ground vehicle's future path is made available to the optimizer. The look ahead algorithm is shown in Algorithm 2. The value of N is determined by multiplying the mesh frequency ($Freq$) by the look ahead time (t_{look}). The look ahead method starts at the UAV's initial conditions (Equation 4.1). For a given look ahead time, the ground vehicle's future location is assumed known, allowing Algorithm 2 to be solved in the same way as Algorithm 1. The resulting optimal control solution is only for the specified look ahead time. The UAV travels along that optimal path solution for Δt seconds until it reaches the next node. Here, the values of the optimal path at this node become the new initial conditions (Equation 4.1). Using the same look ahead time, the process is repeated to find the next optimal path corresponding to the new initial conditions. This process is continually repeated for each node until the node corresponding to $t_f - t_{look}$ is reached. The algorithm cannot use the full time, because it is dependent on future ground vehicle path knowledge. Instead, the final time minus the look ahead time serves as the new final time for the look ahead algorithm. It is important to note that the new optimal path could be resolved at some multiple of Δt if more computational time is required. This means that the UAV travels along each look ahead optimal path for multiple time steps before calculating the new path. The optimal path generated through the look ahead method is an accumulation of numerous, smaller optimal paths. Figure 4.5 shows how these smaller optimal paths are combined to yield the whole optimal path for the ground vehicle's route.

Algorithm 2 : Look Ahead Algorithm

Determine total number of nodes: $N = t_{look} \times Freq$

Set initial conditions (Eqn 4.1)

for $t : \Delta t : (t_f - t_{look})$ **do**

Acquire ground vehicle path and wind information (used as fixed parameters)

Guess control vector (u)

Propagate state equations forward (Eqns 4.2 - 4.6)

Optimizer:

Find u that minimizes cost function (Eqn 4.7)

Satisfy dynamic constraints (Eqns 4.2 - 4.5)

Satisfy path constraints (Table 3.1)

Set optimal states at 2nd time step as new initial conditions

end

Result: Optimal control and optimal states correspond to the initial conditions of each time step from $t : \Delta t : (t_f - t_{look})$

Each colored line in Figure 4.5 represents the optimal path corresponding to a 10 second time interval. Although not shown, the ground vehicle is driving the top half of the figure 8 path. The black, dotted line represents the optimal path output by the look ahead solution. Notice in regions where the SUAS is actively turning, there is a strong variability in the optimal paths from node to node. In other instances, the optimal paths from node to node lie directly on top of each other. The regions of high variability highlight the areas where the full path method has a distinct advantage and will find smoother, more optimal paths. Conversely, when the look ahead optimal paths lie directly on top of each other, the full path solver will converge to a similar, if not identical, solution.

For the look ahead strategy, it is important to find a reasonable time that the optimizer is allowed to "look ahead". There is a palpable tradeoff between the look ahead time and the realism of the model. In reality, the future path of the ground vehicle is not known with perfect certainty and therefore an estimate is required.

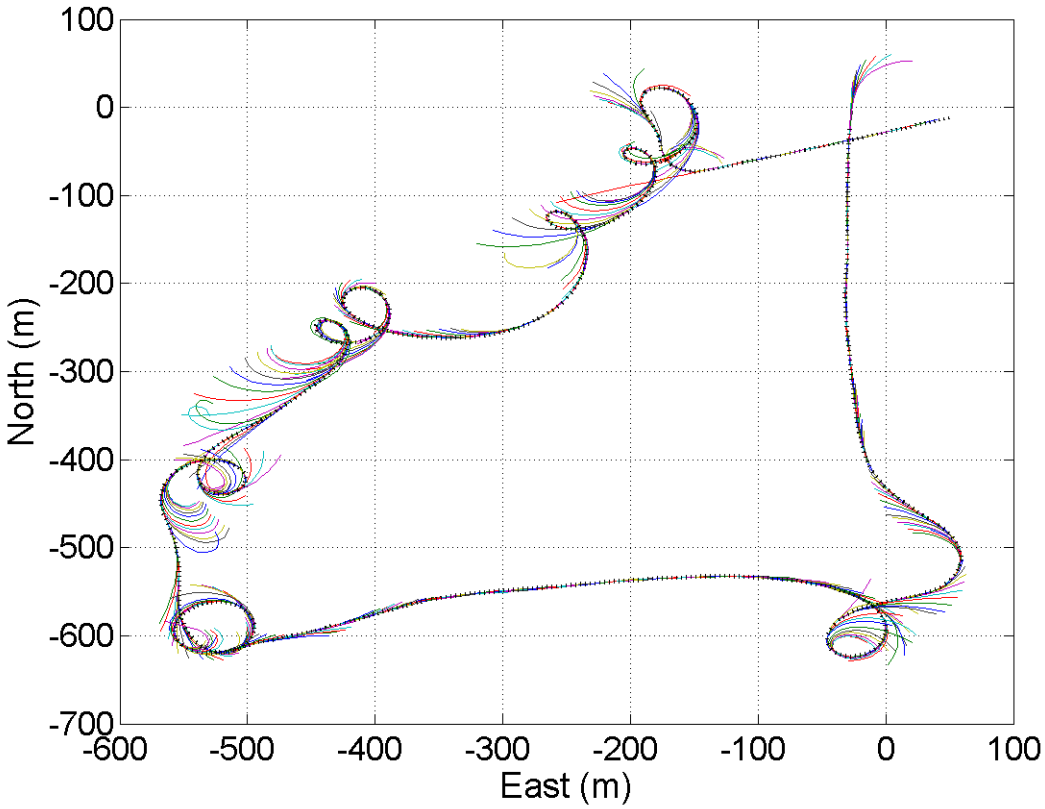


Figure 4.5: Accumulation of Optimal Solutions Found with Look Ahead Algorithm ($t_{look} = 10sec$)

The shorter the look ahead time, the less uncertainty there is in the estimate of the ground vehicle's location. Conversely, longer look ahead times typically result in lower cost function values but are less feasible because of the increased uncertainty related with predicting the ground vehicle's path farther into the future. To determine the appropriate look ahead time, several different look ahead times are plotted and compared in Figure 4.6. The total run time of the ground vehicle is 6.32 minutes. For each look ahead time, Table 4.2 displays the corresponding cost, mesh frequency, number of nodes, total run time and the average run time for each time step.

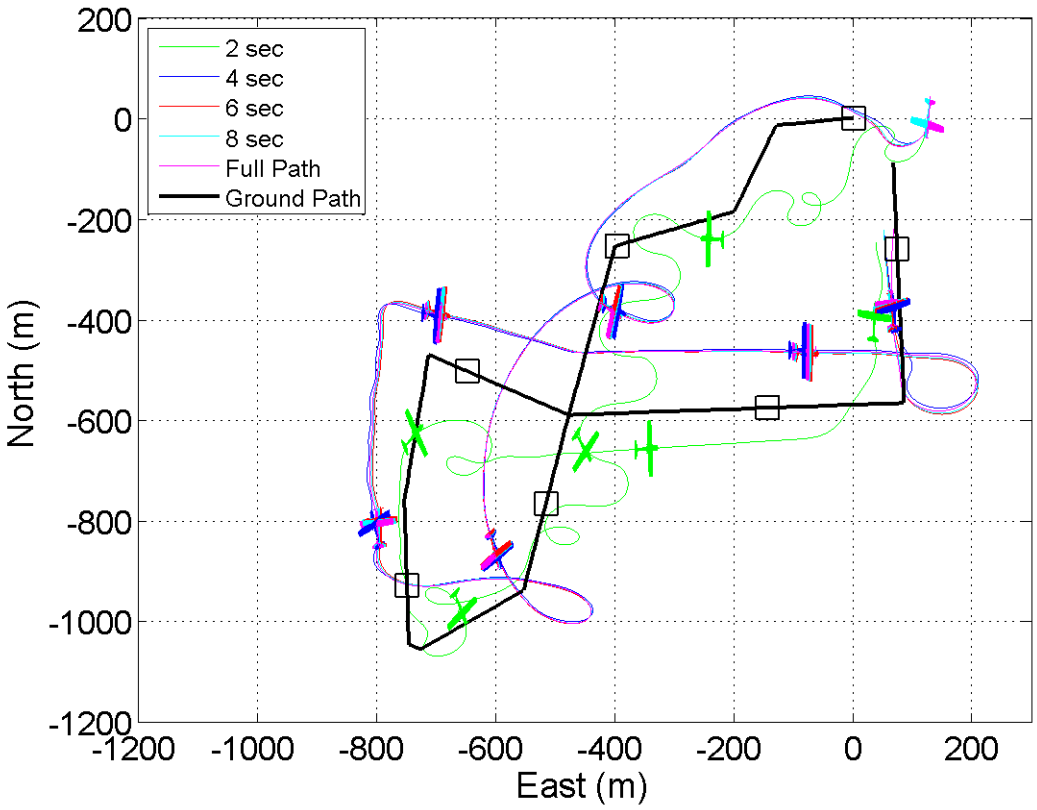


Figure 4.6: Comparison of the Different Look Ahead Times

Table 4.2: Optimization Results for Figure 4.6

Look Ahead Time (sec)	**Cost (J)**	**Mesh Freq (Hz)**	**# of Nodes**	**Run Time (min)**	**Avg Run Time at each Time Step (sec)**
2	47.12	1.5	3	1.24	0.13
4	0.098	1.5	6	1.63	0.17
6	0.075	1.5	9	2.31	0.25
8	0.071	1.5	12	3.06	0.31
Full Path	0.067	1.5	569	30.76	——

When evaluating the flight paths in Figure 4.6, there is a clear difference between the optimal path calculated with a 2 second look ahead and the paths created using a 4 second, 6 second, 8 second and full path method. Table 4.2 shows that the cost of the 2 second solution is over 480 times greater than all of the other solutions. From the start, the 2 second solution immediately diverts from the other four paths. Given the initial starting position and orientation, 2 seconds does not provide sufficient future knowledge for the optimizer to account for the ground vehicle behavior. However, a look ahead time of 4 or more seconds gives sufficient future knowledge for this specific path. In fact, the 4 second, 6 second, 8 second and full path solutions nearly lie on top of each other. This finding is critical because it shows that the added accuracy of the higher look ahead times and the full path method do not result in different optimal paths. The close proximity of the various optimal path solutions suggests that the solution space near the global minimum is flat. The convergence criteria of the optimizer are not specific enough to discriminate between the solutions near the optimal path. This explains why the optimal paths look nearly identical in Figure 4.6 but have slightly different cost values. A flat solution space is desirable because it allows minor deviations to still result in the true optimal path.

The accuracy of the solution is important, but the timeliness of the solution is equally important. Recall from Chapter II, Terning [23] found that Zollar's [25] optimal solution was too slow to effectively work onboard a real-world UAV. Therefore, he developed an approximation that could operate in a timely manner, while still delivering accurate results. The benefit of the look ahead method is that it potentially bridges the gap between the full path optimization and the heuristic-based method. The mesh frequency used for the optimal is 1.5 Hz, meaning that new information is required every 0.667 seconds. Therefore converging on an optimal path must occur at a rate faster than the mesh frequency to be considered as a viable solution. As a general rule of thumb, speeds twice as fast as the mesh frequency are desired because it allows time for the autopilot to implement the controls. Therefore, the look ahead times ranging from 4-8 seconds constitute viable solutions because of their speed and accuracy. This means that optimal controls have the potential to be used for determining the optimal path onboard the real-world, time-sensitive environment of a UAV.

Five other scenarios are investigated to verify the conclusion about the look ahead times and optimality made from Figure 4.6. Figure 4.7 considers five different ground vehicle paths and plots the cost function values for a range of look ahead times. The primary purpose of this plot is to evaluate how look ahead time correlates to optimality. The ground vehicle only drives the top portion of the figure 8 for paths 1 and 5, the bottom portion of the figure 8 for path 2 and the whole figure 8 for paths 3 and 4. For each path, the desired ground vehicle standoff distance is set at 150m. Each path also features different initial conditions, which affect the heading, roll angle and starting location of the UAV (Equation 4.1).

In Figure 4.7, a red circle is shown around the lowest cost function value for each respective path. Every path shows a trend where the jump from 4 to 6 seconds results in the most significant drop in the cost. Interestingly, the disparity in cost between look ahead times of 2 and 4 seconds is at least an order of magnitude greater than the difference between 4 and 6 seconds. For Path 1, J = 21.04 at 2 seconds

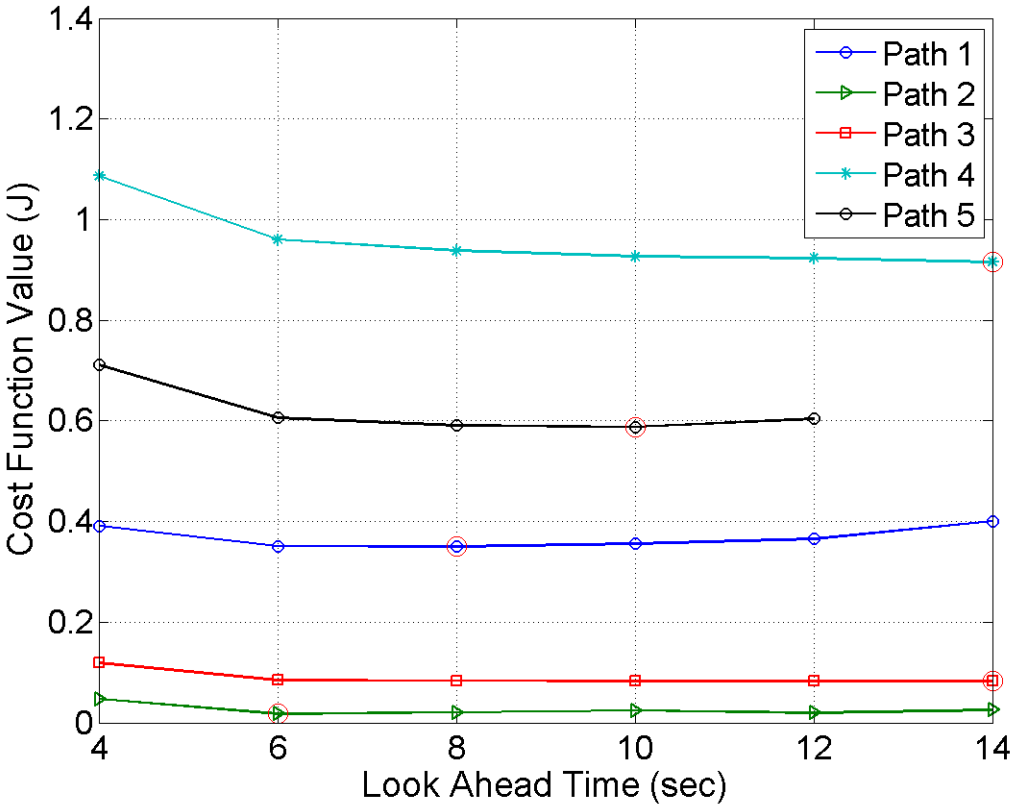


Figure 4.7: Cost Comparison for Different Look Ahead Times

and J = 0.4 at 4 seconds. This large disparity between the 2 second and 4 second look ahead times confirms the similar findings from Figure 4.6 and Table 4.2. The 2 second look ahead time is not plotted because it is substantially larger than the other cost values, thereby obscuring the evaluation of the larger look ahead times relative to each other. Figure 4.7 shows that to converge on a path close to the optimal path, the look ahead time must be at least 4 seconds long. As the look ahead time increases past 6 seconds there is only marginal, oftentimes negligible, improvements in the cost function value, indicating convergence on the optimal path. Figure 4.7 successfully verifies that the conclusions made from analyzing the results shown in Figure 4.6.

Figure 4.8 illustrates how different solvers and settings can lead to slightly different results. First, the 10 second look ahead solution is calculated for the given ground path using the Interior Point Method (IPM). This solution is the initial guess for the full path solver which also uses the IPM. While the solver starts at the initial guess provided by the 10 second look ahead, it iterates and converges to another solution that has a slightly higher cost function value. This same process is repeated using the Active Set Method (ASM) instead of the IPM solver. ASM is another solver available for *fmincon* to use in the solution process. The nuances of ASM are not discussed, but it is important to notice that the choice of the NLP solver can have an effect on the solution. For more information about both the IPM and ASM solvers see reference [15]. The optimal path profiles for the 10 second look ahead, IPM full path and ASM full path methods are shown in Figure 4.8. The initial conditions for all three paths are the same as the UAVs fly to maintain the desired 45m standoff. The aircraft and squares representing the UAV and ground vehicle are plotted at each 60 second interval. Additionally, Table 4.3 displays the cost and run time for each method.

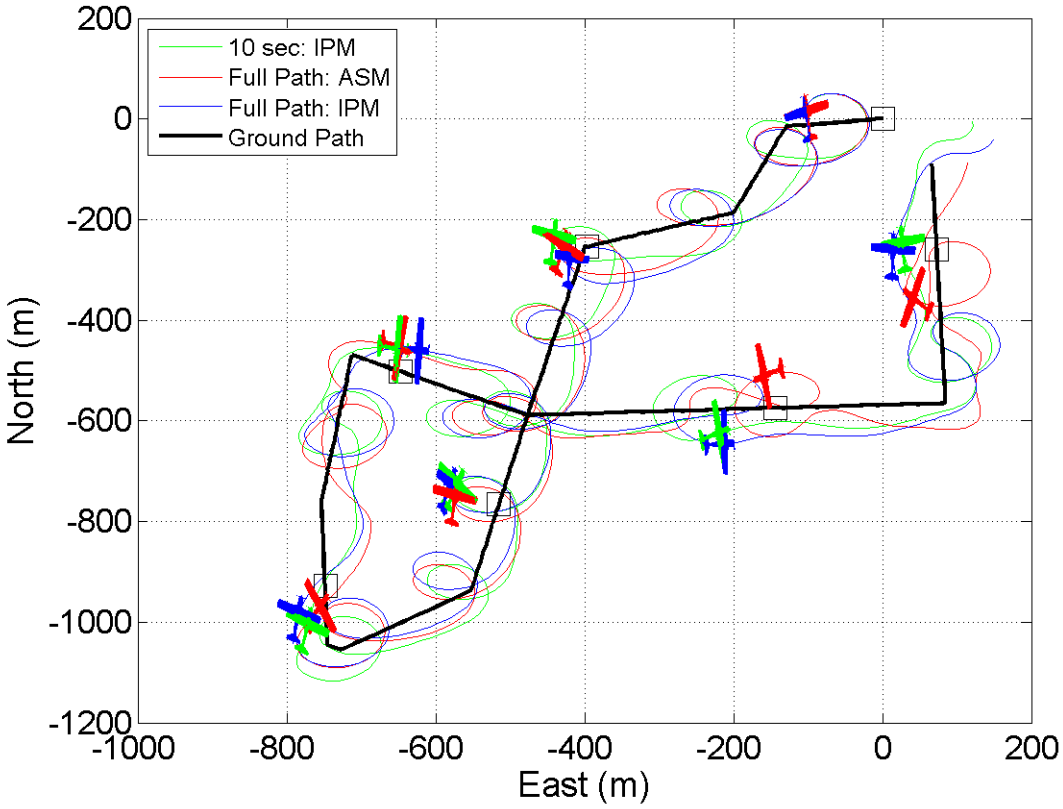


Figure 4.8: Comparison of the Different Optimal Solvers

Table 4.3: Optimization Results for Figure 4.8

Method	**Cost (J)**	**Run Time (min)**
10 sec (IPM)	9.47	7.34
Full Path (ASM)	7.0	828.45
Full Path (IPM)	10.58	64.3

The important takeaway from Figure 4.8 is that the different solution methods result in similar optimal paths. This behavior validates the idea that the solution space near the minimum point is flat. The slight differences in the optimal paths can be attributed to the different convergence tolerances and solution methods of the two optimizers. Enforcing stricter convergence criteria would force all of these methods to the true global minimum point, but at the cost of computational efficiency. The close proximity of the three optimal paths to each other in Figure 4.8 signifies that the convergence criteria is appropriately set for the autonomous convoy overwatch.

Looking back to Figure 4.6, the full path, 4, 6 and 8 second look ahead functions all converged on the optimal path, while the 2 second solution converged to a noticeably different solution. Based on Figure 4.6, there is no advantage in using a full path approach as opposed to the look ahead. The lack of value of the full path algorithm is due largely to weight factor of $\alpha = 0.95$. At this high weight factor, the driving emphasis of the cost function is to minimize the slant range error. This results in little tradeoff between control and slant range error within the cost function. Therefore, the UAV only needs a small amount of future knowledge of the vehicle's location to accurately fly the optimal path. The similarity between the full path and look ahead solutions at $\alpha = 0.95$ allows the team to use a 10 second look ahead time as the new optimal path function. Using the look ahead method is significantly faster and 10 seconds is used as the look ahead time to ensure that the solution safely converges to the solution space containing the global minimum.

Changing the value of α potentially alters the optimal path generated by the look ahead strategy. As α becomes closer to 0.5, the likelihood of a tradeoff between control and slant range error increases. This higher tradeoff probability increases the value of the full path strategy and prevents the look ahead strategy from converging to the optimal solution. The dynamic optimization is sensitive to all potential changes to the cost function, and changing the weight factor changes how the full path and look ahead methods relate to each other. Increasing the vehicle motion and driving a more dynamic path could also cause scenarios where the look ahead method does not converge on optimal solution. Figures 4.6 and 4.8 show that despite different initial conditions and desired loiter ranges, the look ahead strategy sufficiently converges to a local minimum in the solution space for an $\alpha = 0.95$.

The similarity between the look ahead and full path solutions has significant implications on a real-world application. For this given ground path, a full path solution is identical to the look ahead generated path. This is beneficial because using a look ahead method greatly diminishes the computational cost for computing the optimal path. Additionally, modeling the predicted trajectory of the ground vehicle is feasible for a short time range using an estimation routine, such as a Kalman filter. One of the conclusions from evaluating Figure 4.7 is that 4 seconds represents the threshold look ahead time for most ground vehicle paths driving the figure 8. Predicting the vehicle's future performance with a Kalman filter becomes easier as the look ahead time decreases. The conclusion in Chapter VI discusses this application more thoroughly and recommends areas for future development.

The first objective of this thesis was to develop an optimal path for any given ground vehicle path. The algorithm used to determine this optimal path was developed by discretizing the equations of motions and cost function. Two algorithms, the full path and look ahead methods, were used to solve the optimal control problem. The full path method required a priori knowledge of the entire vehicle's path and was significantly more computationally intensive. The look ahead method only required a short amount of future vehicle knowledge and was formed by combining a series of

optimal paths. For the weight value selected, $\alpha = 0.95$, there was little tradeoff between the two objectives in the cost function. This meant that the optimal path using the look ahead method, with a time greater than 4 seconds, converged to the same optimal solution as the full path method. These findings open up the possibilities of using optimal control methods for real-world applications.

The solution strategy developed in this chapter is used for the comparison of the flight test results in Chapter V. The heuristic-based approximation, discussed in Chapter III, is used to approximate the optimal paths created in Chapter V. The look ahead method is used to generate timely optimal paths from the flight test telemetry and its results are compared the actual path flown by the UAV during flight test.

V. Flight Test Results

This chapter achieves the second objective of this thesis effort through comparing the flight tested, heuristic-based approximation with its corresponding optimal path. The test results are accumulated from three separate flight test events spanning from August to December 2013. The purpose of the flight tests are to evaluate how the different iterations of the heuristic-based approximation compare with the optimal path. This chapter breaks down the three flight test events as the initial "follow-me" flight, the intermediate DOE flights and the final flight which uses the parameter values selected from the DOE.

5.1 Follow-me Flight Test

The purpose of the follow-me flight test is to demonstrate the viability of using the APM 2.5 for autonomous UAV tracking of a moving ground vehicle. The notion of optimality or adherence to a desired path is not the focus of this test. Rather, it is essential to develop a working algorithm that the team can further build upon for future iterations. The initial vehicle tracking function implemented onboard the UAV is called the follow-me mode. This mode constantly updates the ground vehicle's location as the desired the loiter point that the SUAS navigates towards. For the follow-me mode, the parameters discussed in Chapter III are not modified from the default autopilot values native to the APM 2.5.

The flight test results, shown in Figure 5.1, compare the performance of the flight tested follow-me mode with its corresponding optimal path. The UAV tries to maintain a 150 m loiter around the moving ground vehicle as it drives the figure 8 ground path. During the flight test, the vehicle waits at the start point to allow for the UAV to loiter around the stationary ground vehicle. The test starts when the UAV is behind the ground vehicle at its 7 o'clock position. At this point, the UAV is turning into the direction that the ground vehicle is driving and is temporarily behind the vehicle. This orientation maximizes the autopilot's options and does not put the UAV in a situation where it must aggressively maneuver to maintain the

desired slant range. A good set of initial conditions are important because they allow for the team to better evaluate how the UAV behaves as the ground vehicle drives the path. If the UAV spends most of its time catching up to the ground vehicle, it diminishes the utility of the test results. The optimal path is generated using the same initial conditions and also incorporates the same wind speed, wind direction and UAV airspeed measured by the autopilot during the flight test. The goal is to create an optimal path that incorporates all of the specific test conditions experienced by the SUAS during the flight test. Every 60 seconds an aircraft is plotted to represent the location of the UAV for both the optimal and flight test paths and a square to represent the ground vehicle location.

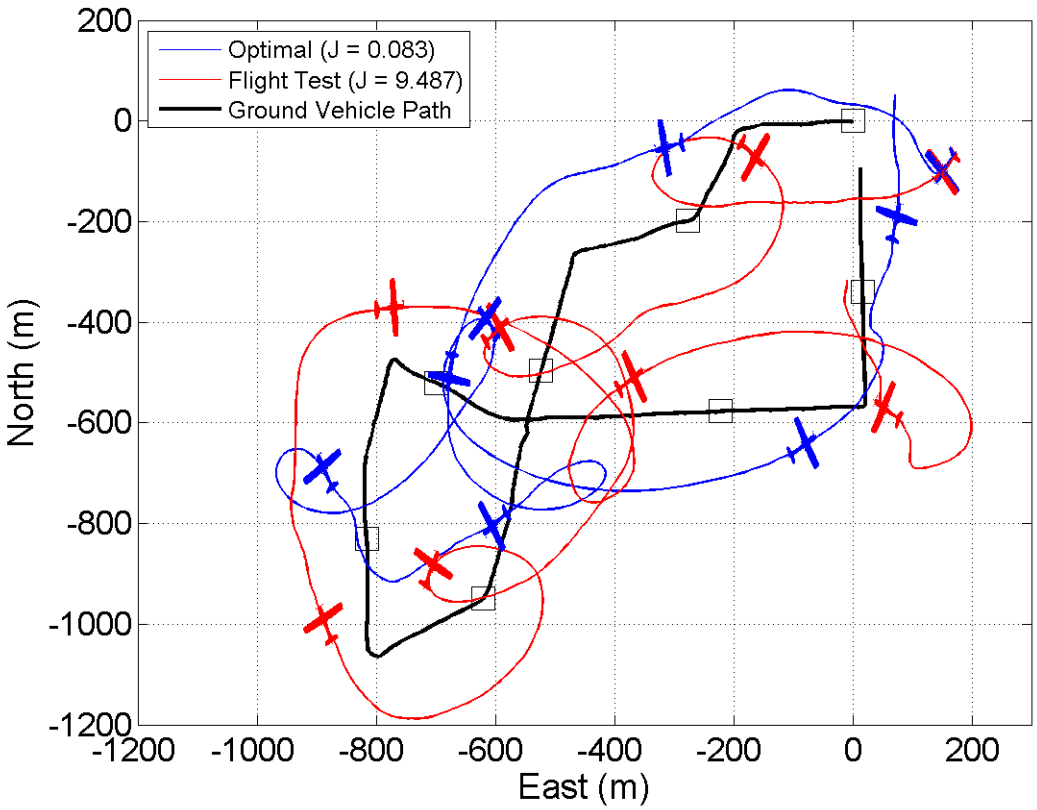


Figure 5.1: Follow-me Flight: Path Comparison

Comparing the performance between the two paths reveals drastic differences and inadequacies of the follow-me mode. The values of the cost function, shown in

the legend, clearly defines how poor the autopilot's performance is compared with the optimal path. The cost function value for the flight test of the follow-me method is almost 113 times greater than the cost of the optimal path. Not only do the cost functional values greatly differ, but there is a substantial difference in the two paths. At the beginning the two paths diverge immediately, with the optimal path going outside and the follow-me path navigating inside of the ground vehicle's path. The two paths remain distinctly separate for the entire ground vehicle path. This significant disparity between the optimal and follow-me paths can be largely attributed to the simplicity of the follow-me mode.

The environmental conditions played a role in diminishing the performance of the follow-me mode. During this flight, the UAV experienced wind speeds close to 65% of its airspeed. Although the identical wind conditions are considered in the optimal path, the presence of strong winds affects how the real-world UAV tracks the moving ground vehicle. The wind data measured by the UAV telemetry system and the reported weather at Camp Atterbury during the flight test are displayed in Table 5.1. The proximity of the telemetry data to the reported weather validates the wind data collected by the UAV. Accurate wind telemetry is important because it makes the optimal path representative of the best possible real-world performance.

Table 5.1: Wind Conditions During Flight Test

	Telemetry	**Reported Weather**[1]
Average Wind Speed	5.75 m/s	5.14 m/s
Average Wind Gusts	9.49 m/s	7.73 m/s
Average Wind Direction	317.8°	315°
Average UAV Airspeed	13.1 m/s	—

It is difficult to discern how well the UAV maintains its desired slant range by looking at Figure 5.1. For this reason, Figure 5.2 compares the slant range of the

[1]Weather for Camp Atterbury, IN on 12/5/2013 at 1115 (using www.wunderground.com)

optimal path versus the flight test path as a function of time. The time intervals are set at 60 seconds to correlate with the aircraft locations in Figure 5.1. This gives perspective on the performance of the UAV over the entire flight path.

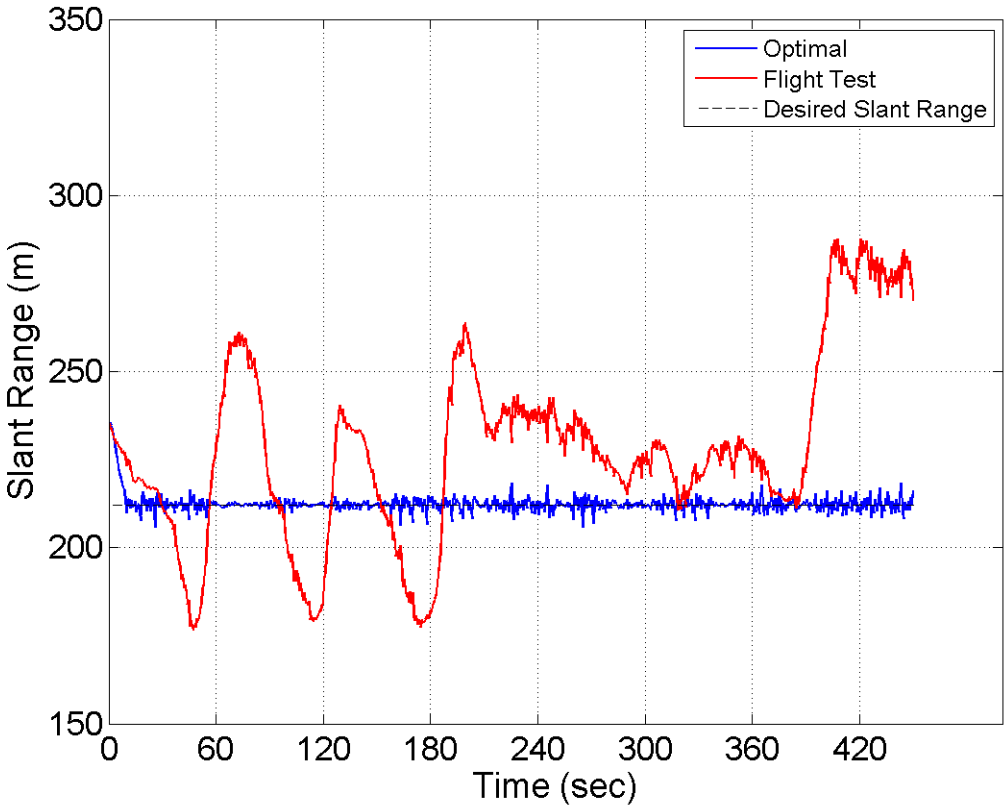


Figure 5.2: Follow-me Flight: Slant Range vs Time

Using the follow-me mode, the UAV successfully tracks and points the camera continuously at the moving ground vehicle. However, there is a substantial difference between the optimal path and the path created by the follow-me mode. This makes sense because the optimal path specifically minimizes the weighted sum of the slant range error and control effort, while the follow-me mode only seeks to loiter about a moving point. However for a fixed altitude and loiter radius, the slant range is fully specified. For the first 240 seconds, the UAV continually overshoots the desired 212 m slant range (150 m loiter radius) as it aggressively tries to maintain the desired standoff. The following 120 seconds show the UAV flying closer to the desired slant

range with less overshoots. This region of performance occurs when the ground vehicle turns to head east and drives straight for 800 m. Coincidentally, the subtle slant range deviation defining this period occurs when the UAV predominantly has a tail wind. The benefit of the tail wind for this period degrades the UAV's performance after the ground vehicle makes the final turn. The ground vehicle's final left-hand turn occurs while the UAV begins a right-hand loiter. Immediately before the turn, the UAV is at the desired standoff, but immediately following the turn its slant range error jumps significantly. To continue loitering around the ground vehicle, the UAV must turn back into the wind. The head wind drastically reduces the UAV's groundspeed, causing the sudden increase in slant range error. There are no spikes in the optimal path because the future location of the ground vehicle is available to the optimizer. Sudden changes of the ground vehicle's velocity are considered when converging to the optimal path.

The initial flight test successfully demonstrates the follow-me autopilot function. However, this method insufficiently approximates the optimal path. Although the high wind speeds exacerbate the autopilot's performance, the two orders of magnitude difference between the cost functions highlights the need for a better approximation.

5.2 DOE Flight Tests

The DOE flight tests are used to determine the parameter values necessary to find the best approximation of the optimal path. The three parameters isolated from the APM default autopilot code are the loiter radius, loiter range and lead time. Each play a distinct role in navigating the SUAS as it loiters around the moving ground vehicle. However, the relative magnitude and interdependence of each parameter is not known. The test matrices, shown in Tables 3.2 and 3.3, specifically highlight the DOE performed to determine the relative values of each parameter to best approximate the optimal path.

Due to some unforeseen setbacks during flight test, all of the test points in the DOE could not be flight tested. Therefore, for consistency, both the test matrices

in Tables 3.2 and 3.3 were flown in HIL. While not ideal, using the HIL simulation allowed the team to use the identical ground vehicle path, the same stochastic model for the wind conditions and test the actual autopilot code. This allowed the cost of the flights to be compared with one another to determine which set of parameters correlated with the most "optimal" performance. The first stage experimental design, with the corresponding costs for each of the test points, is shown in Table 5.2.

Table 5.2: First Stage Experimental Results [17]

Test	**Loiter Range (m)**	**Loiter Radius (m)**	**Lead Time (sec)**	**Cost (J)**
1	182	183	8.9	**4.32**
2	40	125	5.0	**9.65**
3	58	67	8.9	**26.0**
4	120	125	0	**55.48**
5	120	125	5.0	**44.78**
6	58	183	8.90	**7.75**
7	120	125	5.0	**46.97**
8	200	125	5.0	**148.39**
9	182	67	8.9	**111.48**
10	182	183	1.1	**7.06**
11	182	67	1.1	**143.78**
12	120	125	10.0	**47.53**
13	58	183	1.1	**9.10**
14	120	50	5.0	**50.45**
15	120	200	5.0	**29.02**
16	58	67	1.1	**109.75**

Neal [17] runs an analysis of variance (ANOVA) using these data points to determine the value ranges for the loiter range, loiter radius, and lead time that

are statistically relevant for best approximating the optimal path. The result of his analysis is that higher values for loiter radius and lower values for loiter range correlate with improved performance of the heuristic-based approximation. The lead time parameter is deemed statistically insignificant. To converge on the best parameter settings, the second stage experimental design is run in HIL. This test focuses on the regions with a high loiter radius and low loiter range with finer granularity. The cost values for each test point are shown in Table 5.3.

Table 5.3: Second Stage Experimental Results [17]

Test	**Loiter Range (m)**	**Loiter Radius (m)**	**Cost (J)**
1	40	152	**19.29**
2	20	150	**17.19**
3	62	130	**25.15**
4	40	130	**30.08**
5	40	130	**21.44**
6	20	110	**27.74**
7	60	110	**22.61**
8	18	130	**31.88**
9	40	108	**21.57**
10	60	150	**23.25**

The second stage experiment purposefully evaluates the regions with high loiter radius and low loiter ranges. The test point with the loiter range of 20 m and loiter radius of 150 m resulted in the lowest cost. Therefore, the parameter values chosen for the final flight test are a loiter range of 20 m, loiter radius of 150 m and a lead time of 0 seconds. The full analysis of how the values for each parameter are determined is extensively detailed in Neal's [17] work. He explains how he develops the DOE,

chooses the range of values for each parameter and shows all of the regression models for finding the best parameters that approximate the optimal path.

5.3 Final Flight Test

The goal of the final flight test is to use the data gathered from the DOE HIL tests, apply it to the follow-me mode and evaluate the autonomous tracking performance. Additionally, the autopilot code is altered to allow for both right-hand (clockwise) and left-hand (counter-clockwise) loiters. The APM default autopilot code, used for the initial follow-me test, only permits right-hand loiters, which can restrict how the UAV navigates around the moving ground vehicle. The same figure 8 path is driven for this final flight to compare its performance with the follow-me flight.

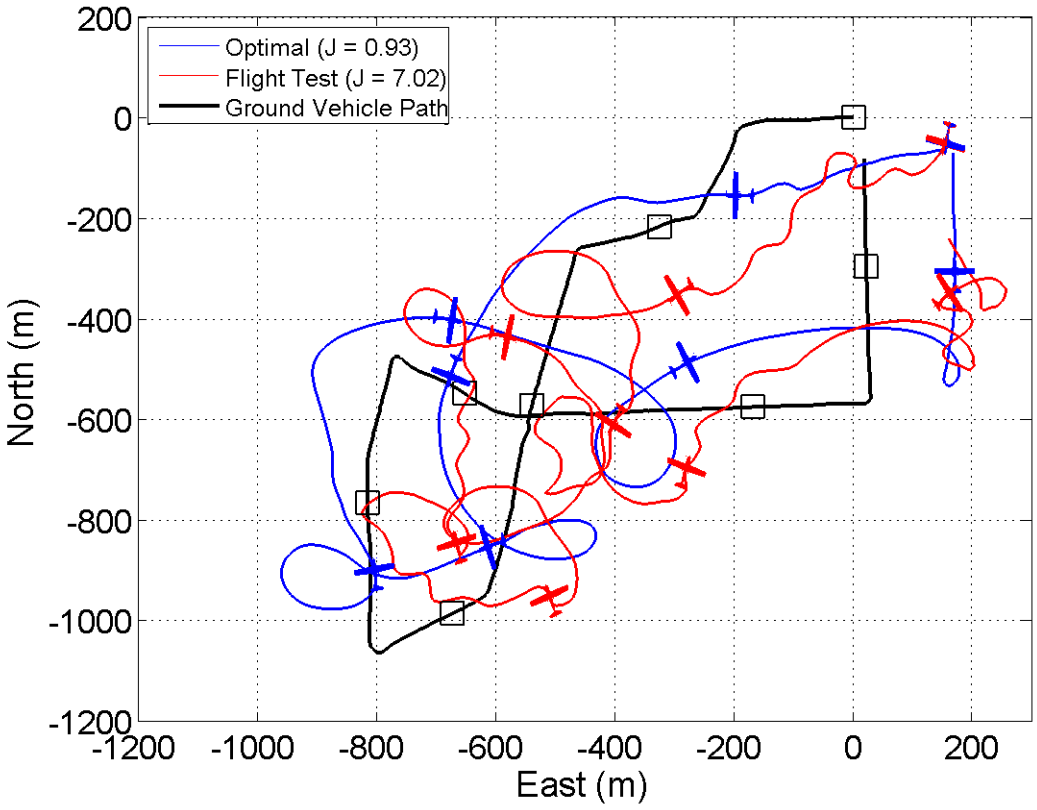


Figure 5.3: Final Flight: Path Comparison

The optimal and flight test paths for the final flight test are shown in Figure 5.3. Identical to Figure 5.1, the optimal path has the same initial conditions as the flight test. The actual airspeed, wind speed and wind direction are also considered when computing the optimal path. The aircraft are plotted every 60 seconds to give a time reference for the entire path flight.

The respective cost function values are the first quantities analyzed when comparing the optimal path to the flight test. The flight test cost function is only 7.5 times greater than the optimal path. The disparity is still significant but the autopilot performance has improved an order of magnitude compared to the initial follow-me. The lower cost function value for the flight test makes intuitive sense by visually comparing the two ground paths. From the start, the autopilot's actions align with the optimal path. While the optimal path performs smoother and smarter maneuvers, the actual flight test features more sporadic motion. This sporadic motion is attributed to the high wind speeds present during the test (Table 5.1) and it negatively impacted the UAV's tracking performance.

The larger cost function value of the optimal path largely contributes to reducing the cost function ratios. Shown in Figure 5.4, the optimal path's initial slant range deviation from the desired is greater than the actual flight test. This deviation can be attributed to the initial conditions and the optimizer's multi-objective considerations. Further complicating the problem is the fact that the UAV has a direct headwind for the first 60 seconds. The headwind significantly reduces the UAV's ground speed and results in the UAV not achieving the desired slant range until the first turn. The optimal solution eventually converges to the desired slant range, although it has a small, oscillatory response instead of consistently holding the desired slant range. These overshoots occur at a frequency of 0.75 Hz, exactly half of the mesh frequency. The dynamic nature of the UAV and the ground vehicle results in small overshoot of the optimal solution at each node. This small noise results in roughly 1 m overshoots and does not destabilize with time. The noise of the optimal solution could be reduced by using a higher mesh frequency, although that would require a more time-intensive

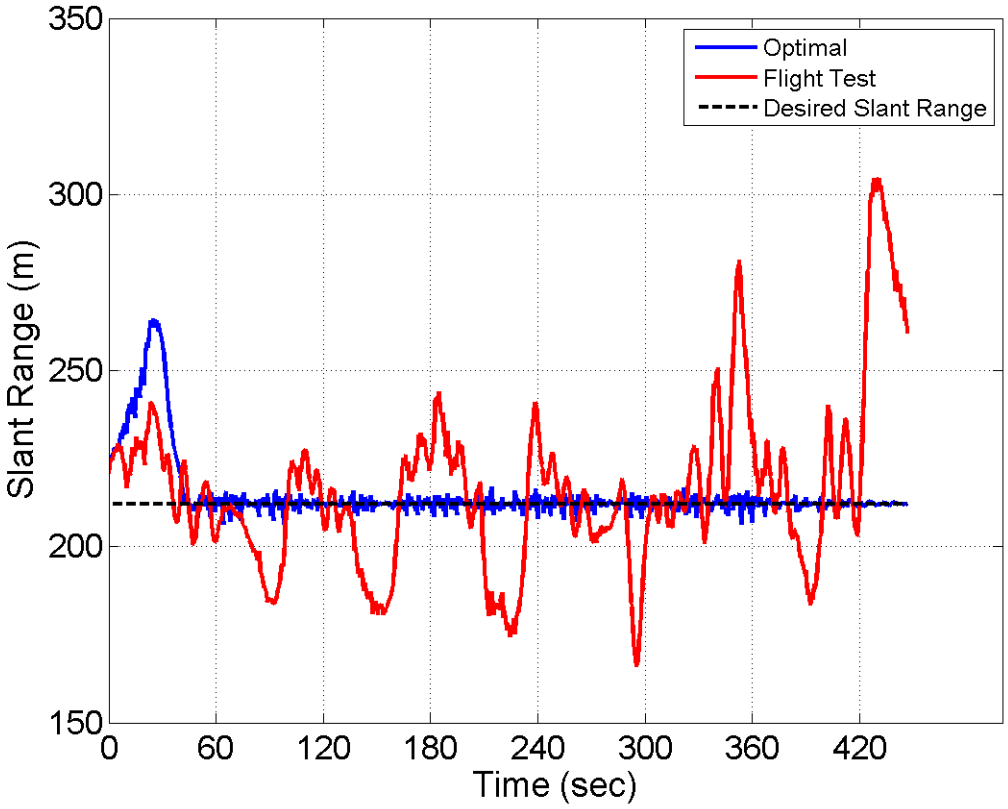


Figure 5.4: Final Flight: Slant Range vs Time

solution process. This result is a by-product of discretizing the mesh using a constant time interval.

The cost of the optimal solution is not only higher, but the heuristic-based approximation does a much improved job of adhering to the desired slant range compared to the follow-me mode. Several slant range overshoots are still seen in Figure 5.4, but they are attenuated and less frequent when compared to the follow-me method. Interestingly, most of the large deviations from the desire slant range occur immediately after the ground vehicle turns. The optimal path is able to anticipate the ground vehicle's turns and proactively maneuver to maintain the desired slant range. The heuristic-based method does not factor any future path knowledge into its path planning algorithm and therefore is more susceptible to slant range deviations following ground vehicle turns.

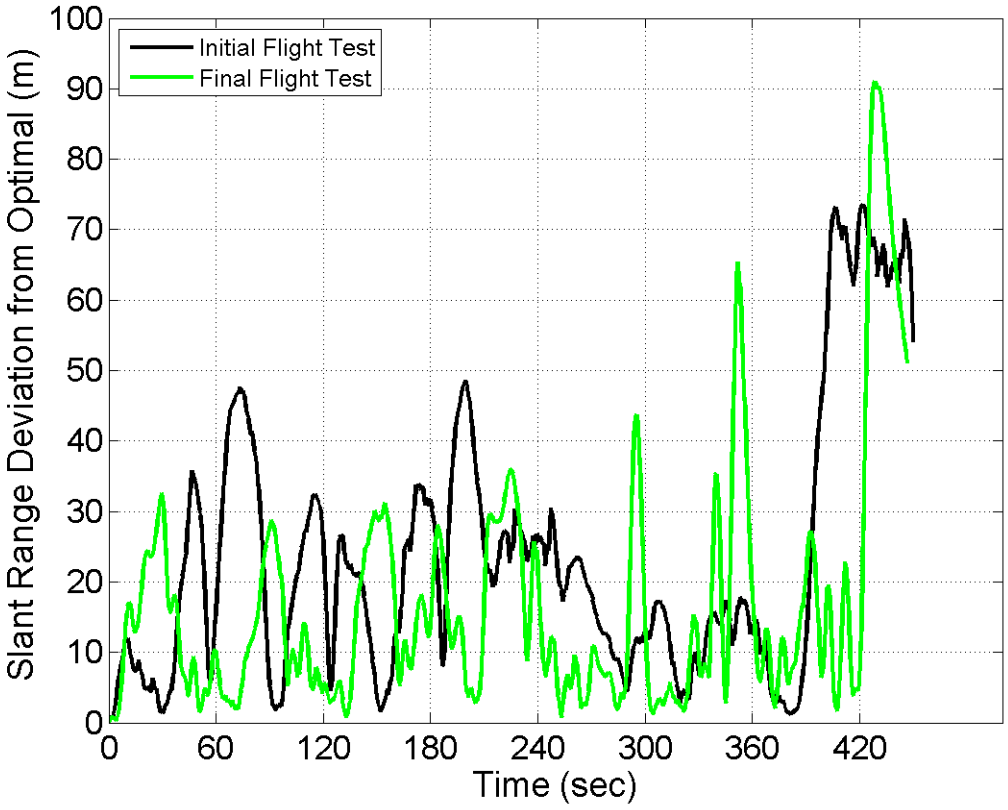


Figure 5.5: Follow-Me vs Final Flight: Slant Range vs Time

The magnitudes of the slant range deviation from the optimal path for both the follow-me and final flight tests are shown in Figure 5.5. Both these flights were conducted within minutes of each other to ensure similar environmental conditions. Although the winds, shown in Table 5.1, were severe, the relative performance of the two methods still gives insight on their ability to approximate the optimal solution. The plots from Figures 5.2 and 5.4 cannot be simply compared because they are based on different ground profiles. By differencing the flight test results from their corresponding optimal paths, the two independent tests can be compared relative to their deviation from the optimal path. The average slant range deviation of the initial flight test is 23.7 m, while the final flight test has an average deviation of 16.9 m. Looking at Figure 5.5, the follow-me flight test has larger peaks and spends less time near the desired slant range (0 m of deviation). The final flight test is not devoid of large peaks; however, it has larger regions where the deviation remains small, especially compared with the initial flight test. The spikes in both flight tests are a result of the ground vehicle turning and are inevitable with the current heuristic method because of its lack of future path knowledge.

The performance of the final flight test is better, but it does not look as improved as the cost function ratio difference of 113 to 7.5 might imply. The ground vehicle paths for the follow-me mode and the final flight are not identical, although they were driven in as close a manner as possible. If the ground paths are assumed identical, then the cost values of the two flight tests can be directly compared. In this case, the final heuristic approximation represents a 26% reduction of the follow-me cost. This improvement is more readily observed in Figure 5.5 than the order of magnitude disparity displayed by the cost value ratios. These results are encouraging, although additional flight tests are needed before any definitive conclusion can be made.

This chapter addressed the second goal by demonstrating an autonomous convoy overwatch algorithm onboard a UAV. The flight test results of the heuristic-based approximations are compared with their corresponding optimal paths. The development of the heuristic-based approximation used a series of flight tests to mature from

a simple follow-me mode to the determine the best parameters to use for the final flight. This process resulted in dropping the cost function ratio from 113 to 7.5. These results are encouraging, although the small sample size of flight test data prevents any general conclusions from being made regarding the two modes. A summary of these results and the path for future work is discussed in Chapter VI.

VI. Conclusion and Recommendations

6.1 Summary

The DoD has a vested interest in developing UAV autonomy and desires to perform autonomous convoy overwatch using SUAS platforms. Specifically, AFRL's intent is to further promote the growth and utility of autonomous systems to complement the warfighter. This thesis aims to support the intent of the DoD, Air Force and AFRL through improving the autonomous capabilities of UAVs. The two objectives of this thesis research were developing an optimal UAV flight path for a given ground vehicle path and approximating that optimal path in a real-world flight test, through heuristic-based methods.

Two algorithms were used to determine the optimal path. They were the full path algorithm, which relied on complete, a priori future path knowledge, and the look ahead algorithm, which was a compilation of optimal paths, each generated by using a small look ahead time (t_{look}). For both cases, the optimizer used a multi-objective cost function that simultaneously sought to minimize the slant range error and the control effort. Section 1.3 lists seven assumptions that were made to help simplify the problem. Most of the assumptions were simplifications that increased UAV endurance (Assumptions 2 and 3), ignored factors with a negligible impact (Assumptions 7) or further constrained the problem (Assumption 5 and 6). Modeling the UAV as a point mass (Assumption 1) and knowing the future location of the ground vehicle (Assumption 4) were the biggest assumptions made in this research. Neglecting the UAV's moments of inertia by modeling it as a point mass significantly simplified the problem. The dynamics of the point mass were constrained to mimic the dynamics of the flight test UAV, creating a more realistic model. Additionally, perfect future knowledge of the ground vehicle's path was a pivotal assumption that allowed the optimal control process to be used.

The full path and look ahead algorithms allowed for the optimal path to be determined based on any ground vehicle path and UAV initial conditions. Both algorithms were discretized into a finite number of equidistant steps. A mesh frequency

of 1.5 Hz was determined to be the best discretization frequency because it was large enough to accurately capture the vehicle motion, while sparse enough to not overburden the space with needless nodes. The optimal path algorithm was solved using the `Matlab`® function *fmincon*. The Interior Point Method (IPM) was selected as the solver to use within *fmincon* because of its ability to scale to complex problems and solve for the optimal control directly.

An analysis was performed to find the best weight factor to use for scaling the multi-objective cost function. A weight factor of $\alpha = 0.95$ was determined to be the best value because it had the best slant range performance with a minor increase in roll rate. At $\alpha = 0.95$, the optimal path found through the full path method was nearly identical to that found by using a look ahead time of 4 or more seconds. The similarities between the full path and look ahead methods highlight the presence of a flat solution space surrounding the global minimum. The conclusion from Chapter IV was that as long as the look ahead or full path solution converged to this solution space containing the global minimum, then it was a sufficient result. This was important because it allowed the look ahead strategy, the more efficient and realistic approach, to be used for computing the optimal paths, with negligible degradation of optimal performance.

A heuristic-based approximation of the optimal path was required for flight test because the autopilot could not run the optimizer onboard the UAV. The first accomplishment was demonstrating an autonomous UAV convoy overwatch functionality using the APM 2.5 autopilot. The autopilot continually updated the location of the moving ground vehicle as the center of its new desired loiter. With the help of Neal [17], the loiter radius, loiter range and lead time were the three parameters used for constructing the heuristic-based approximation. The intermediary flight tests used a series of flights to analyze the interdependence and effects of these three parameters. The outcome of those tests was that the loiter radius should be maximized, the loiter range minimized and that the lead time, as it was defined, did not have a statistically significant impact. Therefore, a loiter radius of 150 m, loiter range of 20 m and

a lead time of 0 seconds, were all determined from the DOE to be the best values to use in the final flight test. The purpose of the final flight test was to determine the improvements of the DOE selected parameters when compared to the follow-me mode.

Overall, the parameters found from the DOE increased the performance of the UAV compared with the initial follow-me mode. The ratio between the flight test and optimal cost function value for the follow-me mode was nearly 113, while the ratio for the final flight was 7.5. This substantial jump in optimality between the two approximations was largely contributed to the increased ability to adhere to the desired slant range of 212 m. The high winds for both the initial and final flight test did not help the UAV's performance, although these environmental conditions were considered in their respective optimal paths. More tests are required to increase the confidence in the performance of the DOE parameters. It is difficult to assess if the performance of the DOE parameters, used in the final flight, sufficiently models the optimal path without specific user requirements and tolerances. Clearly, the flight test path is not close to emulating the optimal path; however, the UAV successfully follows the ground vehicle and maintains 100% time on target. This platform provides a feasible solution for the autonomous convoy overwatch and presents a framework for further increasing its optimality.

6.2 Recommendations

While the objectives of this research were accomplished in this thesis, there are several areas that would benefit from further research. For future work, the main recommendation is to continue to decrease the cost disparity between the optimal path and any heuristic-based approach. This goal is achievable through developing a better future path estimator, creating a more efficient and robust optimizer, improving the optimality of the heuristic-based approach and validating the cost function choices with empirical evidence.

Perfect path knowledge is the biggest assumption made when calculating the optimal path. The ability of the optimizer to predict the future vehicle behavior has a significant effect because it allows the UAV to anticipate ground vehicle changes in speed and direction. Realistically, this type of information is impossible to fully predict; however, there are a few recommended areas for improvement. Kalman filters are mathematical estimators that use a variety of factors to anticipate the future location of the ground vehicle. This technology has matured significantly and has been used in a variety of UAV related applications ([10] and [14]). Allowing the UAV to have access to the convoy routes presents another opportunity for improving the future path prediction functionality. During flight test, sharp ground vehicle turns had the greatest impact on the optimality of the heuristic-based approximation. If these turns were pre-programmed as part of a route plan, the heuristic-based method would better anticipate the ground vehicle maneuvers and more closely adhere to the optimal solution. One or both of these attempts at better predicting the future location of the ground vehicle is necessary to increase the optimality of the heuristic-based method.

Using a more robust optimal control solver is recommended for achieving more accurate results for the optimal path. The difficulty with dynamic optimization is that different initial conditions, paths and constraint boundaries all determine how the solver converges on the optimal path. There are optimal control solvers that use adaptive meshes to efficiently place the nodes for maximum coverage in high gradient areas, while sparsely populating low gradient regions. By smartly placing the nodes, the optimizer can converge on the optimal solution in minimal time. A potential goal is to place an optimal controller onboard the UAV and to be effective, it must be both accurate and timely. The value of a more robust optimizer, coupled with the look ahead method, is that it could meet both these required objectives and provide a viable option for an onboard optimal controller.

The heuristic-based approximation, used in this thesis, showed promise for increasing the optimality of the UAV's path planning function. Neal [17] completed

detailed research to find the best parameter values required to approximate the optimal path. One recommendation would be to continue modifying the existing APM code to continually better approximate the optimal path. Using statistical analysis, coupled with flight test data, is essential for determining the impact the parameters have on the optimal path approximation. The next recommended step is to infuse state-based logic into the heuristic-based method. This means that the UAV's path planning function is dependent not only on the parameters chosen, but the behavior of the ground vehicle and the UAV. Determining how the UAV should behave, while in different "states", will increase the robustness of the approximation and improve its optimality. The key component for this recommendation is more real-world flight test.

The cost function used in this research is contingent on the slant range error and UAV roll rate. The first recommendation is to better understand how these two objectives impact the UAV's performance. Finding the correlation between roll rate and aircraft endurance is recommended to better define the weight factor, α. The lack of information regarding the control input's impact on the UAV's flight endurance led to low consideration in the cost function. This may not necessarily best represent the truly optimal path to fly and deserves better consideration. Investigating the impact of roll rate might also lead to choosing a different control metric that better represents the Sig Rascal's performance. The goal when defining the "optimal" path is to find the path to fly that optimizes the UAV's endurance and adhesion to a given slant range. There is a high measure of confidence in the cost function chosen for this research; however, validating the choices, specifically for the control input, is important to validate the cost function decision.

Considering the gimbal orientation in the cost function is another way of shaping the optimal path. While the UAV ISR mission is stated, it is not explicitly investigated in this thesis. Depending on the ISR mission and platform used, there could possibly be a desired elevation or azimuth angle that would shape the optimal path. The desired elevation or azimuth angle could be added to the cost function to help force

the UAV to not only maintain a desired slant range with minimal control, but also fly to best achieve that angle. Additionally, there could be parts of the UAV (i.e. wings and landing gear) that interfere with the camera's FOV at certain azimuth and elevation angles. Incorporating interference regions into the optimizer's constraints would shape the optimal path to prevent these undesirable camera orientations from happening. All of these additions would be determine from user requirements and would allow the optimization process to be tailored to the specific UAV.

The ultimate goal is to incorporate optimal controls into how unmanned platforms execute the mission. The convoy scenario presents a specific military need that greatly benefits from autonomous UAV capabilities. Through determining the optimal path that minimizes the weighted sum of the slant range error and control, the UAV can provide the best possible coverage for the longest time. This research promotes the continual develop of autonomous, unmanned systems through optimal control integration.

Appendix A. MATLAB Code for Generating Optimal Path

A.1 fmincon Full Path Method

The Matlab® file below is the main script used to generate the optimal path based on a given ground vehicle profile. To successfully run this file, an excel file containing the necessary telemetry data is required. This telemetry file must have the GPS coordinates of the ground vehicle and the UAV, the wind speed and direction, and the UAV's airspeed and roll and yaw angles. The full path method first uses the look ahead method to generate a guess at the optimal control and then uses that output as the initial guess. The two functions called in *fmincon* are the cost function and constraint files and are shown in the subsequent sections.

Listing A.1: PublishMatlab/Final_fullpath.m

```
clc; clear all; close all;
% Inputs:
%-------------------------------------------------------------------------
standoff = 150; % Desired standoff distance / loiter radius
start_time = 0;
final_time = 360; % = End time (from telemetry) - lookahead_time
Telemetry=xlsread('DecFlightDemo.xlsx','State Machine 40m Range','B2:L2992');
Freq = 1.5; %Mesh Frequency
lookahead_time = 10;
set_alpha = 0.95; %Weight factor
%-------------------------------------------------------------------------
t0 = start_time;
tf = t0+lookahead_time;
N = (tf-t0)*Freq;
dt = 1/Freq;
P = pi/180; % shorthand for rad2deg
time_initial = t0:dt:tf;
time = start_time:dt:final_time;
% Telemetry
Ground_track = [Telemetry(:,1), Telemetry(:,6), Telemetry(:,7)];
[Xgv_raw, Ygv_raw] = geod2cart([Ground_track(1,2:3) 220],Ground_track(:,2),...
    Ground_track(:,3));
```

```
time_raw = Ground_track(:,1);
Xgv = interp1(time_raw,Xgv_raw,time_initial);
Ygv = interp1(time_raw,Ygv_raw,time_initial);
%Wind Info
Wind_dir = interp1(time_raw,Telemetry(:,8),time_initial);
Wind_speed = interp1(time_raw,Telemetry(:,9),time_initial);
Vel = interp1(time_raw,Telemetry(:,10),time_initial);
Wind = [Wind_dir;Wind_speed; Vel];
%Flight Test UAV flight path
UAV_track = [Telemetry(:,1), Telemetry(:,2), Telemetry(:,3)];
[Xuav_raw, Yuav_raw] = geod2cart([Ground_track(1,2:3) 220],UAV_track(:,2),...
    UAV_track(:,3));
Xuav_act = interp1(time_raw,Xuav_raw,time_initial);
Yuav_act = interp1(time_raw,Yuav_raw,time_initial);
GV = [Xgv;Ygv];
% Constants: First Loop
Con(1) = 150; %UAV Altitude (m)
Con(2) = 9.81; %Gravitational accel (m/s^2)
Con(3) = sqrt(standoff^2+Con(1)^2); %SR_Desired
Con(4) = 100*P; %umax
Con(5) = set_alpha; %alpha
Con(6) = Freq; %Update Frequency
% Initial Condtions
s0(1) = Xuav_act(1); %Xuav0 (m)
s0(2) = Yuav_act(1); %Yuav0 (m)
s0(3) = interp1(time_raw,Telemetry(:,4),t0); %phi0
s0(4) = interp1(time_raw,Telemetry(:,5),t0); %psi0
% Initialize vectors for optimization
Xuav_test = s0(1)*ones(length(time)-1,N+1);
Yuav_test = s0(2)*ones(length(time)-1,N+1);
phi_opt = s0(3)*ones(1,length(time)-1);
psi_opt = s0(4)*ones(1,length(time)-1);
phi_dot = zeros(1,length(time)-1);
%-------------------------------------------------------------------------
%Control Guess
```

```
u_0 = zeros(1,N+1);
%Constraints on Controls (Rate Limits) degrees*P = radians/second
u1_lb = ones(1,N+1)*-Con(4);
u1_ub = ones(1,N+1)*Con(4);
%Look ahead Optimization
options=optimset('Display','Iter','TolFun',1e-6,'TolX',1e-10,'TolCon',1e-6,'...
    MaxFunEvals',1e6,'MaxIter',2000,'Algorithm','interior-point');
[u_star,cost,FLAG,~,lamda]=fmincon(@(u_0)Thesis_wind_func(u_0,s0,Con,GV,Wind,...
    N,t0,tf),u_0,[],[],[],[],u1_lb,u1_ub,@(u_0)Thesis_wind_cons(u_0,s0,Con,GV,...
    Wind,N,t0,tf),options);
check = 0;
%While statement ensures convergence to an optimal path
while FLAG ~= 1
    check = check+1;
    if FLAG ==0
        disp('Max Iteration Limit Reached');
        pause(5);
        u_0 = u_star;
        [u_star,cost,FLAG,~,lamda]=fmincon(@(u_0)Thesis_wind_func(u_0,s0,Con,...
            GV,Wind,N,t0,tf),u_0,[],[],[],[],u1_lb,u1_ub,@(u_0)...
            Thesis_wind_cons(u_0,s0,Con,GV,Wind,N,t0,tf),options);
        Node(:,check) = [i;FLAG];
    elseif FLAG == 2;
        disp('Change in X is less than options');
        pause(5);
        options2=optimset('Display','Iter','TolFun',1e-6,'MaxFunEvals',1e6,'...
            TolCon',1e-8,'TolX',1e-10,'MaxIter',1000,'Algorithm','interior-...
            point');
        u_0 = u_star;
        [u_star,cost,FLAG,~,lamda]=fmincon(@(u_0)Thesis_wind_func(u_0,s0,Con,...
            GV,Wind,N,t0,tf),u_0,[],[],[],[],u1_lb,u1_ub,@(u_0)...
            Thesis_wind_cons(u_0,s0,Con,GV,Wind,N,t0,tf),options2);
        Node(:,check) = [i;FLAG];
    end
end
```

```
%Find 1st point of the optimal path
[~,Xuav,Yuav,phi,psi] = Thesis_wind_func(u_star,s0,Con,GV,Wind,N,t0,tf);
Xuav_first = Xuav(1);
Yuav_first = Yuav(1);
phi_first = phi(1);
psi_first = psi(1);
phi_dot_first = u_star(1);

for i = 1:length(time)-1
    % New Initial Condtions for each 10sec increment
    s0(1) = Xuav(2); %Xuav0 (m)
    s0(2) = Yuav(2); %Yuav0 (m)
    s0(3) = phi(2); %phi0
    s0(4) = psi(2); %psi0
    t0 = t0+dt; %New initial time
    tf = tf+dt; %New final time
    %-------------------------------------------------------------------
    time_loop = t0:dt:tf;
    if length(time_loop) == N
        time_loop = [time_loop tf];
    end
    %Updated Ground Vehicle location
    Xgv = interp1(time_raw,Xgv_raw,time_loop);
    Ygv = interp1(time_raw,Ygv_raw,time_loop);
    GV = [Xgv;Ygv];
    %Updated Wind Info
    Wind_dir = interp1(time_raw,Telemetry(:,8),time_loop);
    Wind_speed = interp1(time_raw,Telemetry(:,9),time_loop);
    Vel = interp1(time_raw,Telemetry(:,10),time_loop);
    Wind = [Wind_dir;Wind_speed; Vel];
    %-------------------------------------------------------------------
    [u_star,cost,FLAG,~,lamda]=fmincon(@(u_0)Thesis_wind_func(u_0,s0,Con,GV,...
       Wind,N,t0,tf),u_0,[],[],[],[],u1_lb,u1_ub,@(u_0)Thesis_wind_cons(u_0,...
       s0,Con,GV,Wind,N,t0,tf),options);
    %While statement ensures convergence to an optimal path
```

```
        while FLAG ~= 1
116         check = check+1;
            if FLAG ==0
                disp('Max Iteration Limit Reached');
                pause(5);
                u_0 = u_star;
121             [u_star,cost,FLAG,~,lamda]=fmincon(@(u_0)Thesis_wind_func(u_0,s0,...
                    Con,GV,Wind,N,t0,tf),u_0,[],[],[],[],u1_lb,u1_ub,@(u_0)...
                    Thesis_wind_cons(u_0,s0,Con,GV,Wind,N,t0,tf),options);
                Node(:,check) = [i;FLAG];
            elseif FLAG == 2;
                disp('Change in X is less than options');
                pause(5);
126             u_0 = u_star;
                [u_star,cost,FLAG,~,lamda]=fmincon(@(u_0)Thesis_wind_func(u_0,s0,...
                    Con,GV,Wind,N,t0,tf),u_0,[],[],[],[],u1_lb,u1_ub,@(u_0)...
                    Thesis_wind_cons(u_0,s0,Con,GV,Wind,N,t0,tf),options2);
                Node(:,check) = [i;FLAG];
            end
        end
131     %Use the optimal control u_star to find the optimal states
        [~,Xuav,Yuav,phi,psi] =Thesis_wind_func(u_star,s0,Con,GV,Wind,N,t0,tf);
        Xuav_test(i,:) = Xuav;
        Yuav_test(i,:) = Yuav;
        phi_opt(i) = phi(2);
136     psi_opt(i) = psi(2);
        phi_dot(i) = u_star(1);
    end
    % Interpolate the UAV and Ground Vehicle Path over the total time
    Xgv = interp1(time_raw,Xgv_raw,time);
141 Ygv = interp1(time_raw,Ygv_raw,time);
    GV = [Xgv;Ygv];
    Xuav_act = interp1(time_raw,Xuav_raw,time);
    Yuav_act = interp1(time_raw,Yuav_raw,time);
    % Interpolate the Wind information over the total time
```

```
Wind_dir = interp1(time_raw,Telemetry(:,8),time);
Wind_speed = interp1(time_raw,Telemetry(:,9),time);
Vel = interp1(time_raw,Telemetry(:,10),time);
Wind = [Wind_dir;Wind_speed; Vel];
% Roll Rates (Flight Test and Optimal)
phi_dot_act = interp1(time_raw,Telemetry(:,11),time);
phi_dot_opt = [phi_dot_first phi_dot];
%%
%Full Path optimization
clc;
t0 = start_time;
tf = final_time;
N = floor((tf-t0)*Freq);
% Initial Condtions
s0(1) = Xuav_act(1); %Xuav0 (m)
s0(2) = Yuav_act(1); %Yuav0 (m)
s0(3) = interp1(time_raw,Telemetry(:,4),t0); %phi0
s0(4) = interp1(time_raw,Telemetry(:,5),t0); %psi0
%Use the optimal control from look ahead (phi_dot_opt) as initial guess
u_0 = phi_dot_opt;
%Constraints on Controls (Rate Limits) degrees*P = radians/second
u1_lb = ones(1,N+1)*-Con(4);
u1_ub = ones(1,N+1)*Con(4);
%Optimization
u_star=fmincon(@(u_0)Thesis_wind_func(u_0,s0,Con,GV,Wind,N,t0,tf),u_0...
    ,[],[],[],[],u1_lb,u1_ub,@(u_0)Thesis_wind_cons(u_0,s0,Con,GV,Wind,N,t0,tf...
    ),options);
%Use the optimal control u_star to find the optimal states
[J,Xuav_opt,Yuav_opt,phi_opt,psi_opt] = Thesis_wind_func(u_star,s0,Con,GV,...
    Wind,N,t0,tf);
%Camera aimpoint
theta = .01003*phi_opt.^2+.0592*phi_opt+.0268;
aim  = [Xgv-Xuav_opt;Ygv-Yuav_opt;Con(1)*ones(1,length(Xuav_opt))]';
for i = 1:length(aim)
    aimx(i) = aim(i,1)/norm(aim(i,:));
```

```
        aimy(i) = aim(i,2)/norm(aim(i,:));
        aimz(i) = aim(i,3)/norm(aim(i,:));
    end
% Camera Elevation and Azimuth angles
    el = rad2deg(asin((sin(phi_opt).*sin(psi_opt)+cos(phi_opt).*sin(theta).*cos(...
       psi_opt)).*aimx+...
        (-sin(phi_opt).*cos(psi_opt)+cos(phi_opt).*sin(theta).*sin(psi_opt)).*...
           aimy+cos(phi_opt).*cos(theta).*aimz));

    az = rad2deg(atan2(((-cos(phi_opt).*sin(psi_opt)+sin(phi_opt).*sin(theta).*...
       cos(psi_opt)).*aimx+...
        (cos(phi_opt).*cos(psi_opt)+sin(phi_opt).*sin(theta).*sin(psi_opt)).*aimy...
           +sin(phi_opt).*cos(theta).*aimz),...
        (cos(theta).*cos(psi_opt).*aimx+cos(theta).*sin(psi_opt).*aimy-sin(theta)...
           .*aimz)));
    %Slant Range
    SR_opt = sqrt((Xgv-Xuav_opt).^2+(Ygv-Yuav_opt).^2+Con(1)^2);
    SR_act = sqrt((Xgv-Xuav_act).^2+(Ygv-Yuav_act).^2+Con(1)^2);
% Determine the Cost Function
    el_desired = 45*P;
    SR_desired = Con(3);
    umax = Con(4);
    alpha = Con(5);
J_opt = sum((alpha*((SR_opt-SR_desired)/SR_desired).^2+(1-alpha)*(u_star/umax...
       ).^2)*dt);
    J_act = sum((alpha*((SR_act-SR_desired)/SR_desired).^2+(1-alpha)*(phi_dot_act...
       /umax).^2)*dt);
    %% Plot Results
    figure
    plot(Ygv,Xgv,'-k',Yuav_opt,Xuav_opt,'-b',Yuav_act,Xuav_act,'-g');
hold on
    plot(Yuav_opt(1:15:end),Xuav_opt(1:15:end),'bo',Yuav_act(1:15:end),Xuav_act...
       (1:15:end),'gs',Ygv(1:15:end),Xgv(1:15:end),'ko')
    xl = xlabel('East (m)');
    yl = ylabel('North (m)');
```

```
    grid on
206 legend('Ground Vehicle Path','UAV Optimal','UAV Flight Test','Location','...
        NorthWest');
```

A.2 fmincon Cost Function File

This function calculates the cost at each node and reports the total sum. This cost is the value that *fmincon* minimizes through adjusting the control vector.

Listing A.2: PublishMatlab/Thesis_wind_func.m

```
function [J,Xuav,Yuav,phi,psi]=Thesis_wind_func(u1,s0,Con,GV,Wind,N,t0,tf)
%Constants
h = Con(1); %altitude (m)
g = Con(2); %gravity (m/s^2)
SR_desired = Con(3); % desired SR (m)
umax = Con(4); %roll rate max (deg/s)
alpha = Con(5); % weight factor
dt = 1/Con(6); % time interval (sec)
%Wind Speed/Direction
V = Wind(3,:); %airspeed (m/s)
Vw = Wind(2,:);%wind speed (m/s)
psi_w = deg2rad(Wind(1,:)); %wind direction (rad)
%Ground Vehicle Location
Xgv = GV(1,:);
Ygv = GV(2,:);

%Initialize Array/IC's
Xuav = zeros(1,N+1);
Yuav = zeros(1,N+1);
phi = zeros(1,N+1);
psi = zeros(1,N+1);
psi_dot = zeros(1,N+1);
J = zeros(1,N+1);
Xuav(1) = s0(1);
Yuav(1) = s0(2);
phi(1) = s0(3);
psi(1) = s0(4);
psi_dot(1) = g/V(1)*tan(phi(1));
SR = sqrt((Xgv(1)-Xuav(1))^2+(Ygv(1)-Yuav(1))^2+h^2);
J(1) = (alpha*((SR-SR_desired)/SR_desired)^2+(1-alpha)*(u1(1)/umax)^2)*dt;
```

```
for i = 1:N
    Xuav(i+1) = Xuav(i)+(V(i)*cos(psi(i))-Vw(i)*cos(psi_w(i)))*dt;
    Yuav(i+1) = Yuav(i)+(V(i)*sin(psi(i))-Vw(i)*sin(psi_w(i)))*dt;
    phi(i+1) = phi(i)+u1(i)*dt;
    psi_dot(i+1) = g/V(i+1)*tan(phi(i+1));
    psi(i+1) = psi(i)+psi_dot(i)*dt;
    SR = sqrt((Xgv(i+1)-Xuav(i+1))^2+(Ygv(i+1)-Yuav(i+1))^2+h^2);
    J(i+1) = (alpha*((SR-SR_desired)/SR_desired)^2+...
        (1-alpha)*(u1(i+1)/umax)^2)*dt;
end
%Performance Index
 J = sum(J);
```

A.3 fmincon Constraint File

This function defines the path constraints for the system. For any given node, this function prevents any of the path constraints from being violated.

Listing A.3: PublishMatlab/Thesis_wind_cons.m

```
function [g,h] = Thesis_wind_cons(u1,s0,Con,GV,Wind,N,t0,tf)
P = pi/180;
%Constants
h = Con(1); %altitude (m)
g = Con(2); %gravity (m/s^2)
dt = 1/Con(6); % time interval (sec)
%Wind Speed/Direction
V = Wind(3,:); %airspeed (m/s)
Vw = Wind(2,:);%wind speed (m/s)
psi_w = deg2rad(Wind(1,:)); %wind direction (rad)
%Ground Vehicle Location
Xgv = GV(1,:);
Ygv = GV(2,:);
%Initialize Array/IC's
Xuav = zeros(1,N+1);
Yuav = zeros(1,N+1);
phi = zeros(1,N+1);
psi = zeros(1,N+1);
psi_dot = zeros(1,N+1);
el = zeros(1,N+1);

Xuav(1) = s0(1);
Yuav(1) = s0(2);
phi(1) = s0(3);
psi(1) = s0(4);
theta(1) = .1003*phi(1)^2+.0592*phi(1)+.0268;
psi_dot(1) = g/V(1)*tan(phi(1));

aim  = [Xgv(1)-Xuav(1);Ygv(1)-Yuav(1);h];
aimx = aim(1)/norm(aim);
```

```
aimy = aim(2)/norm(aim);
aimz = aim(3)/norm(aim);
el(1) = asin((sin(phi(1)).*sin(psi(1))+cos(phi(1)).*sin(theta(1))...
    .*cos(psi(1))).*aimx+(-sin(phi(1)).*cos(psi(1))+cos(phi(1))...
    .*sin(theta(1)).*sin(psi(1))).*aimy+cos(phi(1)).*cos(theta(1)).*aimz);
for i = 1:N
    Xuav(i+1) = Xuav(i)+(V(i)*cos(psi(i))-Vw(i)*cos(psi_w(i)))*dt;
    Yuav(i+1) = Yuav(i)+(V(i)*sin(psi(i))-Vw(i)*sin(psi_w(i)))*dt;
    phi(i+1) = phi(i)+u1(i)*dt;
    theta(i+1) = .1003*phi(i+1)^2+.0592*phi(i+1)+0.0268;
    psi_dot(i+1) = g/V(i+1)*tan(phi(i+1));
    psi(i+1) = psi(i)+psi_dot(i)*dt;
    aim  = [Xgv(i+1)-Xuav(i+1);Ygv(i+1)-Yuav(i+1);h];
    aimx = aim(1)/norm(aim);
    aimy = aim(2)/norm(aim);
    aimz = aim(3)/norm(aim);
    el(i+1) = asin((sin(phi(i+1)).*sin(psi(i+1))+cos(phi(i+1))...
        .*sin(theta(i+1)).*cos(psi(i+1))).*aimx+(-sin(phi(i+1))...
        .*cos(psi(i+1))+cos(phi(i+1)).*sin(theta(i+1)).*sin(psi(i+1)))...
        .*aimy+cos(phi(i+1)).*cos(theta(i+1)).*aimz);
end
g1 = zeros(1,N+1);
g2 = zeros(1,N+1);
g3 = zeros(1,N+1);
g4 = zeros(1,N+1);
% System Constraints
for i = 1:N+1
    g1(i) = el(i)-80*P; %Max elevation angle
    g2(i) = 0*P-el(i); %Min elevation angle
    g3(i) = phi(i)-40*P; %bank angle (max)
    g4(i) = -40*P-phi(i); %bank angle (min)
end
g = [g1 g2 g3 g4]; %Inequality constraints
h=[]; %Equality constraints
```

Bibliography

1. "DIY Drones: Official ArduPlane Repository", 2013. URL `https://code.google.com/p/ardupilot-mega/wiki/Introduction`.
2. "HackHD", 2013. URL `www.hackhd.com`.
3. "Cloud Cap Technology", 2014. URL `http://www.cloudcaptech.com/`.
4. "Hitec Multiplex", 2014. URL `http://hitecrcd.com/products/servos`.
5. "Kestrel Flight System", 2014. URL `http://www.lockheedmartin.com/us/products/procerus/kestrel.html`.
6. Beard, Randal and Timothy McLain. *Small Unmanned Aircraft: Theory and Practice*. Princeton University Press, Princeton, New Jersey, 2012.
7. Farrell, Shannon. *Waypoint Generation Based on Sensor Aimpoint*. Master's thesis, Air Force Institute of Technology, 2009. AFIT/GAE/ENV/09-M01.
8. Frew, Erik, Dale Lawrence, and Steve Morris. "Coordinated Standoff Tracking of Moving Targets Using Lyapunov Guidance Vector Fields". *Journal of Guidance, Control and Dynamics*, Vol. 31:290–306, 2008.
9. Gertler, Jeremiah. *U.S. Unmanned Aerial Systems*. Technical report, Congressional Research Service, 2012.
10. How, Jonathon, Cameron Fraser, Karl Kulling, Luca Bertuccelli, Olivier Toupet, Luc Brunet, Abe Bachrach, and Nicholas Roy. "Increasing Autonomy of UAVs". *IEEE Robotics and Automation Magazine*, 43–51, 2009.
11. Jodeh, Nidal. *Development of Autonomous Unmanned Aerial Vehicle Research Platform: Modeling, Simulating, and Flight Testing*. Master's thesis, Air Force Institute of Technology, 2006. AFIT/GAE/ENY/06-M18.
12. Jones, Hank. "Mini-UAVs for Convoy Protection". *Unmanned Systems Magazine*, 1–10, 2004.
13. Kim, Il Yong and Olivier de Weck. "Adaptive Weighted Sum Method for Multiobjective Optimization". *AIAA Multidisciplinary Analysis and Optimization Conference*, 2004.
14. Kim, Seungkeun, Hyondong Oh, and Antonios Tsourdos. "Nonlinear Model Predictive Coordinated Standoff Tracking of a Moving Ground Vehicle". *AIAA Guidance, Navigation and Control Conference*, 2011.
15. Lau, Mark, S.P. Yue, K.V. Ling, and J.M. Maciejowski. "A Comparison of Interior Point and Active Set Methods for FPGA Implentation of Model Predictive Control". *European Control Conference*, 156–161, 2009.

16. Lee, Jusuk, Rosemary Huang, Andrew Vaughn, Xiao Xiao, and J. Karl Hedrick. "Strategies for Path-Planning for a UAV to Track a Ground Vehicle". *AINS Conference*, Vol. 2, 2003.

17. Neal, Charles. *Feasibility of Onboard Processing of Heuristic Path Planning and Navigation Algorithm within SUAS Autopilot Computational Constraints*. Master's thesis, Air Force Institute of Technology, 2014.

18. OSD. "Unmanned Systems Roadmap 2007-2032". `https://www.fas.org/irp/program/collect/usroadmap2007.pdf`, 2007.

19. OSD. "Unmanned Systems Integrated Roadmap". `http://www.defense.gov/pubs/DOD-USRM-2013.pdf`, 2013.

20. Quigley, Morgan, Michael Goodrich, Stephen Griffiths, Andrew Eldridge, and Randal Beard. "Target Acquisition, Localization, and Surveillance using a Fixed-Wing Mini-UAV and Gimbaled Camera". *IEEE International Conference of Robotics and Automation*, 2005.

21. Rysdyk, Rolf. "Unmanned Aerial Vehicle Path Following for Target Observation in Wind". *Journal of Guidance, Control and Dynamics*, Vol. 29:1092–1100, 2006.

22. Stolle, Sebastian and Rolf Rysdyk. "Flight Path Following Guidance for Unmanned Air vehicles with Pan-Tilt Camera for Target Observation". *IEEE digital Avionics Systems Conference*, Vol. 2:1–12, 2003.

23. Terning, Nate. *Real-Time Flight Path Optimization for Tracking Stop-and-Go Targets with Micro Air Vehicles*. Master's thesis, Air Force Institute of Technology, 2008. AFIT/GAE/ENY/08-M28.

24. Yoon, Seungho, Sanghyuk Park, and Youdan Kim. "Guidance Law for Standoff Tracking of a Moving Target with Leader-Follower Unmanned Aerial Vehicles". *AIAA Guidance, Navigation and Control Conference*, 1–12, 2012.

25. Zollars, Michael. *Optimal Wind Corrected Flight Path Planning for Autonomous Micro Air Vehicles*. Master's thesis, Air Force Institute of Technology, 2007. AFIT/GAE/ENY/07-M28.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
27–03–2014	Master's Thesis	Oct 2012 — Feb 2014

4. TITLE AND SUBTITLE

Optimal UAV Path Planning for Tracking a Moving Ground Vehicle with a Gimbaled Camera

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Livermore, Riley, A, 2LT, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENY-14-M-33

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFRL/RQQA
2210 8th Street
Bldg 146, Room 300
Wright-Patterson AFB, OH 45433
COMM 937-713-7038
Email: derek.kingston@us.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Distribution Statement A:
Approved for Public Release; Distribution Unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This research develops a path planning algorithm that autonomously controls a UAV to provide convoy overwatch. The optimization algorithm determines the best path to fly through developing a cost function that minimizes the control effort of the UAV and the deviation from a desired slant range. A heuristic-based algorithm was developed and implemented on the autopilot to approximate the optimal solution. In flight test, the UAV successfully tracked a moving ground vehicle by continually placing the UAV's loiter point directly above the ground vehicle's current location. This method was called the "follow-me" mode and provided the baseline for real-world UAV convoy overwatch. The follow-me mode resulted in a cost function value that was 113 times greater than the optimal path. Through an in-depth analysis, the heuristic-based approach reduced this ratio down to only 7.5 times greater than the optimal path. The data collected shows tremendous promise for improving autonomous UAV performance through optimal control techniques.

15. SUBJECT TERMS

Optimal Control, Optimal Path, Path Planning, Cost Function, UAV, SUAS, Target Tracking, Autonomous, Flight Test, Gimbaled Camera

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Richard Cobb, AFIT/ENY
U	U	U	U	118	19b. TELEPHONE NUMBER *(include area code)* (937) 255-3636 x4559; richard.cobb@afit.edu